

# LIS User's Guide

April 30, 2008

Revision 5.0

History:

Revision	Summary of Changes	Date
5.0	Data Assimilation capability	Sept 11, 2006



National Aeronautics and Space Administration  
Goddard Space Flight Center  
Greenbelt, Maryland 20771

History:

Revision	Summary of Changes	Date
4.2	AGRMET supported version	May 11, 2006
4.1	LIS version 4.0.2 release	March 14, 2005
4.0	Milestone "K" submission	February 11, 2005
3.2	LIS version 3.1 release	December 17, 2004
3.1	Milestone "G" release	July 16, 2004
3.0	Milestone "G" submission	May 7, 2004
2.3	Improvements to Milestone "T"	November 30, 2003
2.2	-	-
2.1	Milestone "T" release	November 10, 2003
2.0	Milestone "T" submission	August 14, 2003
1.1	Milestone "F" release	April 25, 2003
1.0	Milestone "F" submission	March 31, 2003

# Contents

<b>1</b>	<b>Introduction</b>	<b>6</b>
1.1	What's New . . . . .	6
1.1.1	LIS 4.2-5.0 . . . . .	6
1.1.2	LIS 4.1-4.2 . . . . .	7
1.1.3	LIS 4.0.2 – 4.1 . . . . .	7
1.1.4	LIS 4.0 – 4.0.2 . . . . .	7
1.1.5	LIS 3.1 – 4.0 . . . . .	7
1.1.6	LIS 3.0 – 3.1 . . . . .	7
1.1.7	LIS 2.0 – 3.0 . . . . .	8
<b>2</b>	<b>Background</b>	<b>9</b>
2.1	LIS . . . . .	9
2.2	LIS core . . . . .	9
<b>3</b>	<b>Preliminaries</b>	<b>11</b>
<b>4</b>	<b>Obtaining the Source Code</b>	<b>12</b>
4.1	Downloading the Source Code . . . . .	12
4.2	Source files . . . . .	12
<b>5</b>	<b>Obtaining the Data-sets</b>	<b>17</b>
5.1	Downloading the Data-sets . . . . .	17
<b>6</b>	<b>Building the Executable</b>	<b>18</b>
6.0.1	Development Tools . . . . .	18
6.0.2	Required Software Libraries . . . . .	18
6.0.3	Optional Software Libraries . . . . .	19
6.0.4	Build Instructions . . . . .	19
6.1	Generating documentation . . . . .	21
<b>7</b>	<b>Running the Executable</b>	<b>22</b>
<b>8</b>	<b>LIS config File</b>	<b>23</b>
8.1	Overall driver options . . . . .	23
8.2	Parameter options . . . . .	26
8.3	Runtime options . . . . .	30
8.4	Data assimilation . . . . .	36
8.5	Domain specification . . . . .	43
8.5.1	Cylindrical lat/lon . . . . .	43
8.5.2	Polar stereographic . . . . .	43
8.5.3	Lambert conformal . . . . .	43
8.5.4	Mercator . . . . .	44
8.5.5	Gaussian . . . . .	44
8.5.6	Polar global . . . . .	44
8.5.7	Cylindrical lat/lon . . . . .	45

8.5.8	Gaussian . . . . .	45
8.5.9	Polar stereographic . . . . .	46
8.6	Parameters . . . . .	46
8.7	Forcings . . . . .	52
8.7.1	GDAS . . . . .	52
8.7.2	GEOS . . . . .	53
8.7.3	ECMWF . . . . .	54
8.7.4	NLDAS . . . . .	55
8.7.5	GSPW . . . . .	57
8.7.6	ECMWF Reanalysis . . . . .	59
8.7.7	AFWA/AGRMET . . . . .	60
8.7.8	NLDAS2 . . . . .	66
8.7.9	PRINCETON . . . . .	67
8.7.10	SALDAS . . . . .	68
8.7.11	Rhone AGG . . . . .	69
8.7.12	GSPW1 . . . . .	70
8.8	Supplimental forcings . . . . .	70
8.8.1	AGRMET radiation . . . . .	70
8.8.2	CMAP precipitation . . . . .	71
8.8.3	CEOP station data . . . . .	71
8.8.4	SCAN station data . . . . .	72
8.8.5	TRMM 3B42RT precipitation . . . . .	72
8.8.6	TRMM 3B42V6 precipitation . . . . .	73
8.8.7	CMORPH precipitation . . . . .	74
8.8.8	Stage II precipitation . . . . .	74
8.8.9	Stage IV precipitation . . . . .	74
8.8.10	D2PCPCAR . . . . .	75
8.8.11	D2PCPOKL . . . . .	75
8.9	Land surface models . . . . .	75
8.9.1	Forcing only – Template . . . . .	75
8.9.2	NCEP’s Noah . . . . .	76
8.9.3	CLM 2.0 . . . . .	78
8.9.4	VIC . . . . .	79
8.9.5	Mosaic . . . . .	82
8.9.6	Hyssib . . . . .	84
8.9.7	SIB2 . . . . .	86
8.9.8	Catchment . . . . .	87
8.10	Model output configuration . . . . .	90
<b>9</b>	<b>Test Cases</b>	<b>93</b>
<b>10</b>	<b>Output Data Processing</b>	<b>94</b>
<b>11</b>	<b>Retrieving Sample Output Data</b>	<b>96</b>
11.1	Downloading the Output Data-sets . . . . .	96
11.2	Example . . . . .	96

<b>A Cylindrical Lat/Lon Domain Example</b>	<b>98</b>
<b>B Polar Stereographic Domain Example</b>	<b>103</b>
<b>C Gaussian Domain Example</b>	<b>104</b>
<b>D Lambert Conformal Domain Example</b>	<b>108</b>
<b>E Mercator Domain Example</b>	<b>109</b>
<b>F Polar Global Domain Example</b>	<b>110</b>
<b>G Makefile</b>	<b>111</b>
<b>H configure.lis</b>	<b>115</b>
<b>I READ GRIB - Information and Instructions</b>	<b>116</b>
I.1 SUBROUTINE GRIBGET (NUNIT, IERR) . . . . .	116
I.2 SUBROUTINE GRIBREAD (NUNIT, DATA, NDATA, IERR) .	117
I.3 SUBROUTINE GRIBHEADER (IERR) . . . . .	117
I.4 SUBROUTINE GRIBDATA (DATARRAY, NDAT) . . . . .	117
I.5 SUBROUTINE GRIBPRINT (ISEC) . . . . .	118
I.6 SUBROUTINE GET_SEC1 (KSEC1) . . . . .	118
I.7 SUBROUTINE GET_SEC2 (KSEC2) . . . . .	118
I.8 SUBROUTINE GET_GRIDINFO (IGINFO, GINFO) . . . . .	118
I.9 C-ROUTINE COPEN (UNIT, NUNIT, NAME, MODE, ERR, OFLAG) . . . . .	118
I.10 SEC Header Array Information Tables . . . . .	119
I.11 Additional information for setting up the READ_GRIB routines for use on Linux Machines . . . . .	121
I.12 Example of Fortran code that calls READ_GRIB routines . . . . .	122
<b>J GRIB Output Information</b>	<b>125</b>
J.1 SUBROUTINE GRIB1_SETUP . . . . .	125
J.2 SUBROUTINE GRIB1_FINALIZE . . . . .	126
J.3 SUBROUTINE DRV_WRITEVAR_GRIB . . . . .	127
J.4 Output GRIB Variables . . . . .	127
J.5 Example of Fortran code that calls GRIB output routines . . . . .	129

# 1 Introduction

This is the LIS' User's Guide. This document describes how to download and install the code and data needed to run the LIS executable for LIS revision 5.0. It describes how to build and run the code, and finally this document also describes how to download output data-sets to use for validation.

This document consists of 9 sections, described as follows:

- 1 Introduction:** the section you are currently reading
- 2 Background:** general information about the LIS project
- 3 Preliminaries:** general information, steps, instructions, and definitions used throughout the rest of this document
- 4 Obtaining the Source Code:** the steps needed to download the source code
- 5 Obtaining the Data-sets:** the steps needed to download the data-sets
- 6 Building the Executable:** the steps needed to build the LIS executable
- 7 Running the Executable:** the steps needed to prepare and submit a run, also describes the various run-time configurations
- 10 Output Data Processing:** the steps needed to post-process generated output for visualization
- 11 Retrieving Sample Output Data:** the steps needed to download data generated by the LIS team during its testing runs

## 1.1 What's New

See *RELEASE\_NOTES* found in the *source.tar.gz* file for more details. (See Section 4.)

### 1.1.1 LIS 4.2-5.0

1. This version includes the infrastructure for performing data assimilation using a number of different algorithms from simple approaches such as direct insertion to the more sophisticated ensemble kalman filtering.
2. More streamlined support for different architectures: A configuration based specification for the LIS makefile.
3. The data assimilation infrastructure utilizes the Earth System Modeling Framework (ESMF) structures. The LIS configuration utility has been replaced with the corresponding ESMF utility.

### **1.1.2 LIS 4.1-4.2**

1. Completed implementation of AGRMET processing algorithms
2. Ability to run on polar stereographic, mercator, lambert conformal, and lat/lon projections
3. Updated spatial interpolation tools to support the transformations to/from the above grid projections
4. Switched to a highly interactive configurations management from the fortran namelist-based lis.crd style.
5. Streamlined error and diagnostic logging, in both sequential and parallel processing environments.
6. extended grib support; included the UCAR-based read-grib library
7. Support for new supplemental forcing analyses - Huffman, CMORPH

### **1.1.3 LIS 4.0.2 – 4.1**

1. Preliminary AFWA support
2. Ability to run on a defined layout of processors.
3. Updates to plugins, preliminary implementation of alarms.
4. Definition of LIS specific environment variables.

### **1.1.4 LIS 4.0 – 4.0.2**

1. GSWP-2 support – LIS can now run GSWP-2 experiments. Currently only CLM and Noah models have full support.
2. Updates to the 1km running mode.
3. Updates to the GDS running mode.

### **1.1.5 LIS 3.1 – 4.0**

1. VIC 4.0.5 – LIS' implementation of VIC has been reinstated.

### **1.1.6 LIS 3.0 – 3.1**

1. New domain-plugin support – facilitates creating new domains.
2. New domain definition support – facilitates defining running domains. Sub-domain selection now works for both MPI-based and non MPI-based runs.

3. New parameter-plugin support – facilitates adding new input parameter data-sets.
4. New LIS version of ipolates – facilitates creating new domains and base forcing data-sets.
5. Compile-time MPI support – MPI libraries are no longer required to compile LIS.
6. Compile-time netCDF support – netCDF libraries are no longer required to compile LIS.
7. New LIS time manager support – ESMF time manager was removed. ESMF libraries are not required in this version of LIS.

#### **1.1.7 LIS 2.0 – 3.0**

1. Running Modes – Now there is more than one way to run LIS. In addition to the standard MPI running mode, there are the GDS running mode and the 1 km running mode.
2. Sub-domain Selection – Now you are no longer limited to global simulations. You may choose any sub-set of the global domain to run over. See Section ?? and Section ?? for more details. (This is currently only available for the MPI-based running mode.)
3. Plug-ins – Now it is easy to add new LSM and forcing data-sets into the LIS driver. See LIS' Developer's Guide for more details.

## 2 Background

This section provides some general information about the LIS project and land surface modeling.

### 2.1 LIS

The primary goal of the LIS project is to build a system that is capable of performing high resolution land surface modeling at high performance using scalable computing technologies. The LIS software system consists of a number of components: (1) LIS driver: the core software that integrates the use of land surface models, data management techniques, and high performance computing. (2) community land surface models such as CLM [3], and Noah [5] and (3) Visualization and data management tools such as GrADS [1] -DODS [4] server. One of the important design goals of LIS is to develop an interoperable system to interface and interoperate with land surface modeling community and other earth system models. LIS is designed using an object oriented, component-based style. The adaptable interfaces in LIS can be used by the developers to ease the cost of development and foster rapid prototyping and development of applications. The following sections describe the main components of LIS.

### 2.2 LIS core

The central part of LIS software system is the LIS core that controls program execution. The LIS core is a model control and input/output system (consisting of a number of subroutines, modules written in Fortran 90 source code) that drives multiple offline one-dimensional LSMs. The one-dimensional LSMs such as CLM and Noah, apply the governing equations of the physical processes of the soil-vegetation-snowpack medium. These land surface models aim to characterize the transfer of mass, energy, and momentum between a vegetated surface and the atmosphere. When there are multiple vegetation types inside a grid box, the grid box is further divided into “tiles”, with each tile representing a specific vegetation type within the grid box, in order to simulate sub-grid scale variability.

The execution of the LIS core starts with reading in the user specifications, including the modeling domain, spatial resolution, duration of the run, etc. Section 7 describes the exhaustive list of parameters specified by the user. This is followed by the reading and computing of model parameters. The time loop begins and forcing data is read, time/space interpolation is computed and modified as necessary. Forcing data is used to specify the boundary conditions to the land surface model. The LIS core applies time/space interpolation to convert the forcing data to the appropriate resolution required by the model. The selected model is run for a vector of “tiles” and output and restart files are written at the specified output interval.

Some of the salient features provided by the LIS core include:

- Vegetation type-based “tile” or “patch” approach to simulate sub-grid scale variability.
- Makes use of various satellite and ground-based observational systems.
- Derives model parameters from existing topography, vegetation, and soil coverages.
- Extensible interfaces to facilitate incorporation of new land surface models, forcing schemes.
- Uses a modular, object oriented style design that allows “plug and play” of different features by allowing user to select only the components of interest while building the executable.
- Ability to perform regional modeling (only on the domain of interest).
- Provides a number of scalable parallel processing modes of operation.

Please refer to the software design document for a detailed description of the design of LIS core. The LIS reference manual provides a description of the extensible interfaces in LIS. The “plug and play” feature of different components is described in this document.

### 3 Preliminaries

This section provides some preliminary information to make reading this guide easier.

Commands are written like this:

```
% cd /path/to/LISv5.0  
% ls  
“... compiler flags, the run gmake.”
```

File names are written like this:

*/path/to/LISv5.0/src*

You need to create a working directory on your system to install and run LIS in. Let's call this directory */path/to/LISv5.0/*. Throughout the rest of this document, this directory shall be referred to as *\$WORKING*. You should create a directory to run LIS in. This directory can be created anywhere on your system, but, in this document, we shall refer to this running directory as *\$WORKING/run*.

## **4 Obtaining the Source Code**

This section describes how to obtain the source code needed to build the LIS executable.

The source code is maintained in a Subversion repository; thus, you need the Subversion client (svn) to obtain this code. If you need any help regarding Subversion, please go to <http://subversion.tigris.org/>.

### **4.1 Downloading the Source Code**

To obtain the source code needed for LIS revision 5.0:

1. Go to LIS’ “Public Release Home Page”  
Go to <http://lis.gsfc.nasa.gov/>  
Follow the “Source Codes” link.  
Follow the “LIS 5.0 Code Release” link.
2. From LIS’ “Public Release Home Page”  
Follow the instructions in the “Source Code” section.

### **4.2 Source files**

Checking out the LIS source code (according the instructions in Section 4) will create a directory named *src*. The structure of *src* is as follows:

Directory Name	Synopsis
arch	Directory containing the configurable options for building the LIS executable
baseforcing	Top level directory for base forcing methods
baseforcing/berg	Routines for handling ECMWF reanalysis forcing product
baseforcing/ecmwf	Routines for handling ECMWF forcing product
baseforcing/ecmwfreanal	Routines for handling ECMWF Reanalysis forcing product
baseforcing/geos	Routines for handling GEOS forcing product
baseforcing/gdas	Routines for handling GDAS forcing product
baseforcing/nldas	Routines for handling NLDAS forcing product
baseforcing/nldas2	Routines for handling NLDAS-2 forcing product
baseforcing/saldas	Routines for handling SALDAS forcing product
baseforcing/gswp	Routines for handling GSWP-2 forcing product
baseforcing/gswp1	Routines for handling GSWP-1 forcing product
baseforcing/princeton	Routines for handling Princeton forcing product
baseforcing/rhone	Routines for handling Rhone forcing product
baseforcing/template	Baseforcing template routines
core	LIS core routines
plugins	Modules defining the function table registry of extensible functionalities
domains	Top level directory for domain implementations
domains/latlon	Routines for creating grid and tile spaces in a lat/lon projection
domains/lambert	Routines for creating grid and tile spaces in a lambert conformal projection
domains/merc	Routines for creating grid and tile spaces in a mercator projection
domains/polar	Routines for creating grid and tile spaces in a polar stereographic projection
domains/gswp	Routines for creating grid and tile spaces for a GSWP simulation
domains/catchment	Routines for creating grid and tile spaces in a catchment simulation
domains/gaussian	Routines for creating grid and tile spaces in a Gaussian projection
domains/hrap	Routines for creating grid and tile spaces in a hrapping simulation
interp	Interpolation routines
lib	Libraries needed for linking

lsms/clm2	Top level clm2 land surface model sub-directory
lsms/clm2/biogeochem	Biogeochemistry routines
lsms/clm2/biogeophys	Biogeophysics routines (e.g., surface fluxes)
lsms/clm2/camclm_share	Code shared between the clm2 and cam (e.g., calendar information)
lsms/clm2/csm_share	Code shared by all the geophysical model components of the Community Climate System Model (CCSM). Currently contains code for CCSM message passing orbital calculations and system utilities
lsms/clm2/ecosysdyn	Ecosystem dynamics routines (e.g., leaf and stem area index)
lsms/clm2/main	Control (driver) routines
lsms/clm2/mksrfdata	Routines for generating surface data-sets
lsms/clm2/utils	Independent utility routines
lsms/clm2/cpl_wrf	Routines for coupling CLM with WRF
lsms/mosaic	mosaic land surface model
lsms/noah.2.7.1	Noah land surface model version 2.7.1
lsms/noah.2.7.1/da_snow	Routines for assimilating snow
lsms/noah.2.7.1/da_soilm	Routines for assimilating soil moisture
lsms/noah.2.7.1/da_sca	Routines for assimilating snow cover
lsms/noah.2.7.1/cpl_wrf	Routines for coupling Noah with WRF
lsms/hyssib	hyssib land surface model
lsms/SiB2	SiB2 land surface model
lsms/catchment	Catchment land surface model
lsms/catchment/da_soilm	Routines for assimilating soil moisture
lsms/sacramento	Sacramento land surface model
lsms/snow17	snow17 land surface model
lsms/sacsnow17	Sacramento with snow17 land surface model
lsms/mlbc	mlbc land surface model
lsms/vic	VIC land surface model
lsms/template	templates for incorporating a land surface model

make	Makefile and needed headers
suppforcing	Top level directory for supplemental products
suppforcing/cmap	Routines for handling CMAP precipitation product
suppforcing/cmorph	Routines for handling CMORPH precipitation product
suppforcing/stg2	Routines for handling stage-2 precipitation product
suppforcing/stg4	Routines for handling stage-4 precipitation product
suppforcing/3B42RT	Routines for handling 3B42RT precipitation product
suppforcing/3B42V6	Routines for handling 3B42V6 precipitation product
suppforcing/ceop	Routines for handling CEOP precipitation product
suppforcing/scan	Routines for handling SCAN precipitation product
suppforcing/agrrad	Routines for handle AGRMET radiation product
params	Top level directory for input parameter products
params/topo	Routines for handling topography data
params/lai	Routines for handling leaf/stem area index data
params/landcover	Routines for handling land cover classification data
params/soils	Routines for handling soil classification data
params/tbot	Routines for handling bottom temperature data
params/albedo	Routines for handling albedo data
params/gfrac	Routines for handling greenness fraction data
runmodes	Top level directory for different running modes
runmodes/retrospective	Routines to manage a retrospective mode
runmodes/wrf_cpl_mode	Routines to manage a LIS-WRF coupled mode
runmodes/GAoptimization	Routines to manage an optimization mode
offline	Contains the main program for the offline mode of operation

dataassim	Top level directory for data assimilation support
dataassim/algorithm	Top level directory containing data assimilation algorithms
dataassim/algorithm/di	Routines for data assimilation using direct insertion
dataassim/algorithm/gmaoenkf	Routines for data assimilation using GMAO's Ensemble Kalman Filter
dataassim/perturb	Perturbation algorithm implementations
dataassim/perturb/static	Static gaussian normal perturbation algorithm
dataassim/perturb/gmaopert	GMAO's pertubation algorithm
dataassim/obs	Directory containing various observation data handling for data assimilation
dataassim/obs/syntheticsm1	synthetic soil moisture data handling routines for DA
dataassim/obs/syntheticsm2	synthetic soil moisture data handling routines for DA
dataassim/obs/syntheticswe1	synthetic snow water equivalent data handling routines for DA
dataassim/obs/syntheticsca	synthetic snow cover data handling routines for DA
dataassim/obs/syntheticsca	AMSRE soil moisture data handling routines for DA
configs	some sample LIS configuration files
utils	miscellaneous information
testcases	testcases for verifying various functionalities

Source code documentation may be found on LIS' web-site on LIS' "Public Release Home Page". Follow the "LIS 5.0 Source Code Documentation" link.

## **5 Obtaining the Data-sets**

This section describes how to obtain sample data-sets for testing the LIS executable.

The LIS team has created many test cases for testing various features and functionality of LIS.

Please read Section 9 for more details about the test cases.

### **5.1 Downloading the Data-sets**

To obtain sample data-sets for LIS revision 5.0:

1. Go to LIS’ “Public Release Home Page”  
Go to <http://lis.gsfc.nasa.gov/>  
Follow the “Source Codes” link.  
Follow the “LIS 5.0 Code Release” link.
2. From LIS’ “Public Release Home Page”  
Follow the instructions in the “Input Data” section.

## 6 Building the Executable

This section describes how to build the source code and create LIS' executable – named LIS.

### 6.0.1 Development Tools

This code has been compiled and run on Linux PC (Intel/AMD based) systems, IBM AIX systems, and SGI Altix system. These instructions expect that you are using such a system. In particular you need

- Linux
  - Absoft's Pro Fortran Software Development Kit, version 10.0
  - or
  - Lahey/Fujitsu's Fortran 95 Compiler, release L6.00c
  - GNU's C and C++ compilers, gcc and g++, version 3.3.3
  - GNU's make, gmake, version 3.77
- IBM
  - XL Fortran version 10.1.0.6
  - GNU's make, gmake, version 3.77
- SGI Altix
  - Intel Fortran Compiler version 9.1.0.42
  - GNU's make, gmake, version 3.77

### 6.0.2 Required Software Libraries

In order to build the LIS executable, the following libraries must be installed on your system:

- Earth System Modeling Framework (ESMF) version 2.2.2rp3 (<http://www.esmf.ucar.edu/>)  
Please read the ESMF User's Guide for details on compiling ESMF with MPI support and without MPI support ("mpiuni").

Note that the `ESMF_Config` implementation in `esmf_2_2*` versions have a limitation that may cause a crash when used with LIS 5.0. LIS uses the `ESMF_Config` object for the implementation of runtime specifications through `lis.config` file. The `ESMF_Config` object in `esmf_2_2*` versions are currently limited to 200 lines (excluding comments). If the number of lines in `lis.config` exceeds 200, the program will fail. To correct this, change the following line in the ESMF source code, before generating the library.

Go to `esmf/src/Infrastructure/Config/src/ESMF_Config.F90` change the line:

```
integer, parameter :: MSZ = 200
```

to

```
integer, parameter :: MSZ = 500
```

This will enable the use of *lis.config* files of size upto 500 lines (excluding comments).

### 6.0.3 Optional Software Libraries

The following libraries are not required to compile LIS. They are used to extend the functionality of LIS.

- Message Passing Interface (MPI) – If you wish to run the MPI-based running mode
  - vendor supplied, or
  - MPICH version 1.2.7p1 (<http://www-unix.mcs.anl.gov/mpi/mpich1/>)
- If you choose to have NETCDF support, please download the netcdf source (<http://www.unidata.ucar.edu/software/netcdf/>) and compile the source to generate the NETCDF library.
- If you choose to have HDF support, please download the hdf source (<http://hdf.ncsa.uiuc.edu/>)

To install these libraries, follow the instructions provided at the various URL listed above.

Note: Due to the mix of programming languages (Fortran and C) used by LIS, you may run into linking errors when building the LIS executable.

When compiling code using Absoft's Pro Fortran SDK, set the following compiler options:

```
-YEXT_NAMES=LCS -s -YEXT_SFX=_ -YCFRL=1
```

These must be set for each of the above libraries.

### 6.0.4 Build Instructions

1. Perform the steps described in Section 4 to obtain the source code.
2. Goto the *\$WORKING/src/arch* directory. A number of files named *configure.lis.\** exist in this directory. Each file contains the configurable options that are specific for each architecture and compiler. For example, the file *configure.lis.aix* contains the set of configurable options for an IBM AIX platform. Depending on your choice of platform, edit this file or create a new file for your platform with the set of options. The following is a list of variables that need to be specified in the *configure.lis* file.

Variable	Description
FC	fortran 90 compiler
FC77	fortan 77 compiler
LD	fortran linker
CC	C compiler
INC_NETCDF	path to NETCDF header files
LIB_NETCDF	path to NETCDF library files
INC_HDF	path to HDF header files
LIB_HDF	path to HDF library files
INC_HDFEOS	path to HDFEOS header files
LIB_HDFEOS	path to HDFEOS library files
LIB_MPI	path to mpi libraries
INC_MPI	path to mpi header files
LIB_ESMF	path to esmf libraries
MOD_ESMF	path to esmf modules
CFLAGS	flags for C compiler
FFLAGS	flags for Fortran 90 compiler
FFLAGS77	flags for Fortran 77 compiler
LDFLAGS	flags for linker

If the user chooses to compile and run on a single processor with no MPI, the options in the *configure.lis* file should be specified accordingly. Specifying the compiler preprocessor flag *-DSPMD* enables the compiling of the code with MPI support. Removing this flag produces a serial version of LIS.

3. Compile the new GRIB library, *libw3.a*. You must edit the *Makefile* located in *\$WORKING/src/lib/w3lib*. Uncomment the appropriate block of compiler flags, then run **gmake**.
4. Compile the new GRIB library, *griblib.a*. You must edit the *Makefile* located in *\$WORKING/src/lib/grib*. Uncomment the appropriate block of compiler flags, then run **gmake**.
5. Compile the new GRIB library, *read\_grib*. You must edit the *Makefile* located in *\$WORKING/src/lib/read\_grib*. Uncomment the appropriate block of compiler flags, then run **gmake <arch>**, where *<arch>* is the appropriate architecture or compiler label. Running **gmake** will produce a list of acceptable values. For example, to compile on a system using the Intel Fortran compiler, run **gmake ifc**. Please refer to the Appendix I for helpful suggestions and instructions. If you are on an IBM AIX system, use the *read\_grib.aix* library.
6. All the included libraries are generated. Copy the appropriate *configure.lis.\** file to *\$WORKING/src/make/configure.lis* and edit this *configure.lis* file to make sure the file paths are specified correctly.
7. Compile the dependency generator, *makdep*. Change directory into *\$WORKING/src/make/MAKDEP*. Run **gmake**.

8. Compile the LIS source code.
  - (a) Change directory into  $\$WORKING/src/make$ .  
 $\% cd \$WORKING/src/make$
  - (b) Edit the *misc.h* file to specify if NETCDF support should be included.  
 If `define USE_NETCDF` is set, NETCDF support will be included.  
 To disable NETCDF support, edit the *misc.h* file to specify `UNDEF USE_NETCDF`.
  - (c) Edit the *misc.h* file to specify if HDF support should be included.  
 If `define USE_HDF` is set, HDF support will be included.  
 To disable HDF support, edit the *misc.h* file to specify `UNDEF USE_HDF`.
  - (d) Run the make command.  
 $\% gmake$
  - (e) Finally, copy the *LIS* executable into your running directory,  $\$WORKING/run$ .

See Appendix G to see the *Makefile*. See Appendix H to see a *configure.lis* file.

## 6.1 Generating documentation

LIS code uses the ProTex documenting system [2]. The documentation in LATEX format can be produced by using the `doc.csh` in the  $\$WORKING/src/utils$  directory. This command produces documentation, generating a number of LATEX files. These files can be easily converted to pdf or html formats using utilites such as `pdflatex` or `latex2html`.

## 7 Running the Executable

This section describes how to run the LIS executable. Once the LIS executable is built, a simulation can be performed using the *lis.config* file.

The single-process version of LIS is executed by the following command issued in the *\$WORKING/run/* directory.

```
% ./LIS
```

The parallel version of LIS must be run through an *mpirun* script or similar mechanism. Assuming that MPI is installed correctly, the LIS simulation is carried out by the following command issued from in the *\$WORKING/run/* directory.

```
% mpirun -np N ./LIS
```

The *-np N* flag indicates the number of processes to use in the run, where you replace *N* with the number of processes to use. On a multiprocessor machine, the parallel processing capabilities of LIS can be exploited using this flag.

To customize your run, you must modify the *lis.config* configuration file. See Section 8 for more information.

## 8 LIS config File

This section describes the options in the *lis.config* file.

### 8.1 Overall driver options

**Running mode:** specifies the running mode used in LIS. Acceptable values are:

Value	Description
1	retrospective mode
2	AFWA AGRMET mode
3	Coupled WRF mode
4	Coupled GCE mode
5	Coupled GFS mode

**Running mode:** 1

**Domain type:** specifies the LIS domain used for the run Acceptable values are:

Value	Description
1	Lat/Lon projection with SW to NE data ordering
2	GSPW domain
3	Polar stereographic projection with SW to NE data ordering
4	Lambert conformal projection with SW to NE data ordering
5	Mercator projection with SW to NE data ordering
6	AFWA lat/lon 0.5 degree domain with no subgrid tiling (used for the benchmarks)
7	Reserved, not supported
8	Catchment based domain
9	Gaussian domain
10	HRAP domain

**Domain type:** 1

**Number of nests:** specifies the number of nests used for the run. Values 1 or higher are acceptable. The maximum number of nests is limited by the amount of available memory on the system. The specifications for different nests are done using white spaces as the delimiter. Please see below for further explanations. Note that all nested domains should run on the same projection and same land surface model.

**Number of nests:** 1

**Land surface model:** specifies the land surface model to run. Acceptable values are:

Value	Description
0	template lsm
1	Noah
2	CLM
3	VIC
4	mosaic
5	hyssib
6	sib2
7	catchment
8	reserved
9	reserved
10	reserved
11	reserved
12	reserved
13	reserved

**Land surface model:** 1

**Base forcing source:** specifies the forcing data source for the run. Acceptable values are:

Value	Description
0	template base forcing
1	GDAS
2	GEOS
3	ECMWF
4	NLDAS
5	GWP2
6	ECMWF Reanalysis
7	AGRMET
8	NLDAS2
9	Princeton
10	SALDAS
11	Rhone
12	GWP1

Base forcing source: 1

**Supplemental forcing source:** specifies the supplemental forcing data source for the run. Acceptable values are:

Value	Description
0	None
1	AGRMET Radiation
2	CMAP
3	Reserved
4	CEOP
5	SCAN
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved
11	TRMM 3B42RT
12	TRMM 3B42V6
13	Reserved
14	CMORPH
15	Stage II
16	Stage IV
17	D2PCPCAR
18	D2PCPOKL

Supplemental forcing source: 0

## 8.2 Parameter options

The following options list the choice of parameter maps to be used

**Landcover data source:** specifies the usage of soils data in the run. Acceptable values are:

Value	Description
1	use the UMD landcover
2	use the USGS landcover data
3	use the GFS landcover data
4	use the IGBP landcover data

**Landcover data source:** 1

**Use soil texture:** specifies the usage of soil texture data in the run. Acceptable values are:

Value	Description
0	use a sand/silt/clay percentage map
1	use a texture map

**Use soil texture:** 0

**Soil data source:** specifies the source of soil parameters in the run. Acceptable values are:

Value	Description
1	use FAO based maps
2	use STATSGO based maps
3	use GFS based maps

**Soil data source:** 1

**Soil color data source:** specifies the source of soil color data in the run. Acceptable values are:

Value	Description
0	do not use soil color
1	use FAO based map
1	use STATSGO based map

**Soil color data source:** 0

**Elevation data source:** specifies topography data source for the run. Acceptable values are:

Value	Description
0	do not use
1	GTOPO30 based
2	Reserved
3	GFS based

**Elevation data source:** 1

**slope map:** specifies the slope of the LIS grid. Acceptable values are:

Value	Description
0	do not use
1	GTOPO30 based

**Slope data source:** 0

**aspect map:** specifies the aspect of the LIS grid. Acceptable values are:

Value	Description
0	do not use
1	GTOPO30 based

**Aspect data source:** 0

**curvature map:** specifies the curvature of the LIS grid. Acceptable values are:

Value	Description
0	do not use
1	GTOPO30 based

**Curvature data source:** 0

**LAI data source:** specifies the LAI data source for the run. Acceptable values are:

Value	Description
0	do not use
1	AVHRR-based LAI
2	MODIS-based LAI
3	Reserved
4	Catchment-based LAI

**LAI data source:** 0

**SAI data source:** specifies the SAI data source for the run. Acceptable values are:

Value	Description
0	do not use
1	AVHRR-based SAI
2	MODIS-based SAI
3	Reserved
4	Catchment-based SAI

**SAI data source:** 0

**Albedo data source:** specifies if albedo data is to be used in the run. Acceptable values are:

Value	Description
0	Do not read albedo data
1	NCEP Climatology
2	Reserved
3	GFS

**Albedo data source:** 1

**Greenness data source:** specifies if greenness fraction data is to be used in the run. Acceptable values are:

Value	Description
0	Do not read gfrac data
1	NCEP Climatology
2	Catchment based data
3	GFS

**Greenness data source:** 1

**Porosity data source:** specifies if soil porosity data is to be used in the run.  
Acceptable values are:

Value	Description
0	Do not read porosity data
1	FAO-based data

**Porosity data source:** 0

**Ksat data source:** specifies if hydraulic conductivity data is to be used in the run. Acceptable values are:

Value	Description
0	Do not read ksat data
1	FAO-based data

**Ksat data source:** 0

**B parameter data source:** specifies if the b parameter data is to be used in the run. Acceptable values are:

Value	Description
0	Do not read b parameter data
1	FAO-based data

**B parameter data source:** 0

**Quartz data source:** specifies if the quartz data is to be used in the run.  
Acceptable values are:

Value	Description
0	Do not read quartz data

**Quartz data source:** 0

**Snow data source:** specifies if the snow data is to be used in the run. Acceptable values are:

Value	Description
0	Do not read snow data
1	Read SNODEP data

**Snow data source:** 0

### 8.3 Runtime options

**Experiment code:** specifies the “experiment code number” for the run. It is used in constructing the name of the output directory for the run. Acceptable values are any name using up to 3 characters.

**Experiment code:** '111'

**Number of veg types:** specifies the number of vegetation types used in the landcover data. Acceptable values are:

Value	Description
13	UMD-based landcover types
16	IGBP-based landcover types
30	USGS-based landcover types

**Number of veg types:** 13

**Number of forcing variables:** specifies the number of forcing variables.

**Number of forcing variables:** 10

**Forcing variables list file:** specifies the file containing the list of forcing variables to be used. This list consists of two columns of information – the name of the forcing variable, and its units. This information is listed in the order that the forcing variables are read.

Forcing variables list file: ./input/forcing\_variables.txt

**Number of supplemental forcing variables:** specifies the number of forcing variables in the supplemental forcing.

Number of supplemental forcing variables: 0

**Use elevation correction:** specifies whether to use elevation correction for forcing.

Acceptable values are:

Value	Description
0	Do not use elevation correction for forcing
1	Use lapse rate correction for forcing
2	Use micro met correction for forcing – not supported

Use elevation correction: 1

**Spatial interpolation method:** specifies the type of interpolation scheme to apply to the forcing data. Acceptable values are:

Value	Description
1	bilinear scheme
2	conservative scheme
3	neighbour scheme

Bilinear interpolation uses 4 neighboring points to compute the interpolation weights. The conservative approach uses 25 neighboring points. If the conservative option is turned on, it is used to interpolate the precip field only (to conserve water). Other fields will still be interpolated with the bilinear option.

Spatial interpolation method: 1

**Temporal interpolation method:** specifies the type of temporal interpolation scheme to apply to the forcing data. Acceptable values are:

Value	Description
1	linear scheme
2	ueber next scheme

The linear temporal interpolation method computes the temporal weights based on two points. Ubernext computes weights based on three points. Currently the ubernext option is implemented only for the GSWP forcing.

**Temporal interpolation method:** 1

**Output forcing:** specifies whether to output the ALMA optional forcing variables. Acceptable values are:

Value	Description
0	Do not output forcing variables
1	Do output forcing variables

**Output forcing:** 1

**Output parameters:** specifies whether to output the ALMA optional parameter variables. Acceptable values are:

Value	Description
0	Do not output parameter variables
1	Do output parameter variables

**Output parameters:** 0

**Output methodology:** specifies whether to write output as a 1-D array containing only land points or as a 2-D array containing both land and water points. Acceptable values are:

Value	Description
0	Do not write output
1	Write output in a 1-D grid domain
2	Write output in a 2-D grid domain

**Output methodology:** 2

**Output data format:** specifies the format of the model output data. Acceptable values are:

Value	Description
1	Write output in binary format
2	Write output in Grib format
3	Write output in NETCDF format

**Output data format:** 1

**Logging level:** specifies different levels of logging. Acceptable values are:

Value	Description
1	Default messages
2	Default messages and additional messages
3	Debugging messages

**Logging level:** 1

**Start mode:** specifies if a restart mode is being used. Acceptable values are:

Value	Description
1	A restart mode is being used
2	A cold start mode is being used, no restart file read

When the cold start option is specified, the program is initialized using the LSM-specific initial conditions (typically assumed uniform for all tiles). When a restart mode is used, it is assumed that a corresponding restart file is provided depending upon which LSM is used. The user also needs to make sure that the ending time of the simulation is greater than model time when the restart file was written.

**Start mode:** 2

The start time is specified in the following format:

Variable	Value	Description
Starting year:	integer 2001 – present	specifying starting year
Starting month:	integer 1 – 12	specifying starting month
Starting day:	integer 1 – 31	specifying starting day
Starting hour:	integer 0 – 23	specifying starting hour
Starting minute:	integer 0 – 59	specifying starting minute
Starting second:	integer 0 – 59	specifying starting second
Starting year:	2002	
Starting month:	10	
Starting day:	29	
Starting hour:	1	
Starting minute:	0	
Starting second:	0	

The end time is specified in the following format:

Variable	Value	Description
Ending year:	integer 2001 – present	specifying ending year
Ending month:	integer 1 – 12	specifying ending month
Ending day:	integer 1 – 31	specifying ending day
Ending hour:	integer 0 – 23	specifying ending hour
Ending minute:	integer 0 – 59	specifying ending minute
Ending second:	integer 0 – 59	specifying ending second
Ending year:	2002	
Ending month:	10	
Ending day:	31	
Ending hour:	1	
Ending minute:	0	
Ending second:	0	

**Model timestep:** specifies the time-step for the run. Acceptable values are:

Value	Description
900	15 minute time-step
1800	30 minute time-step
3600	60 minute time-step

For a nested domain, the timesteps for each nest should be specified with white spaces as the delimiter. If two domains (one subnest) are employed, the first one using 900 seconds and the second one using 3600 seconds as the timestep, the model timesteps are specified as:

E.g.: Model timestep: 900 3600

**Model timestep:** 1800

**Undefined value:** specifies the undefined value. The default is set to -9999.

**Undefined value:** -9999

**Output directory:** specifies the name of the top-level output directory. Acceptable values are any 40 character string. The default value is set to OUTPUT. For simplicity, throughout the rest of this document, this top-level output directory shall be referred to by its default name, *\$WORKING/LIS/OUTPUT*.

**Output directory:** 'OUTPUT'

**Diagnostic output file:** specifies the name of run time diagnostic file. Acceptable values are any 40 character string.

**Diagnostic output file:** 'lisdiag'

**Number of ensembles per tile:** specifies the number of ensembles of tiles to be used. The value should be greater than equal to 1.

**Number of ensembles per tile:** 1

The following options are used for subgrid tiling based on vegetation

**Maximum number of tiles per grid:** defines the maximum tiles per grid (this can be as many as the total number of vegetation types).

**Maximum number of tiles per grid:** 1

**Cutoff percentage:** defines the smallest percentage of a cell for which to create a tile. The percentage value is expressed as a fraction.

Cutoff percentage: 0.05

This section specifies the 2-d layout of the processors in a parallel processing environment. The user can specify the number of processors along the east-west dimension and north-south dimension using **Number of processors along x:** and **Number of processors along y:**, respectively. Note that the layout of processors should match the total number of processors used. For example, if 8 processors are used, the layout can be specified as 1x8, 2x4, 4x2, or 8x1.

Number of processors along x: 2  
Number of processors along y: 2

## 8.4 Data assimilation

This section specifies the choice of data assimilation options.

**Assimilation algorithm:** specifies the choice of data assimilation algorithms. Acceptable values are:

Value	Description
0	None
1	Direct Insertion
2	GMAO EnKF data assimilation

Assimilation algorithm: 0

**Assimilation set:** specifies the “assimilation set”, which is the combination of the variable being updated + the observation being assimilated. For example, an assimilation set could be defined as the soil moisture update in Noah land surface model with assimilating an AMSR-E soil moisture dataset. For the example implemented (Noah soil moisture assimilation with a synthetic soil moisture data generated from a previous Noah run).

Acceptable values are:

Value	Description
0	Noah soil moisture assimilation + synthetic data

**Assimilation set:** 0

**Number of state variables:** specifies the number of state prognostic variables used in the assimilation.

**Number of state variables:** 0

**Number of observation types:** specifies the number of operation species/types used in the assimilation.

**Number of observation types:** 0

**Output interval for diagnostics:** specifies the output diagnostics interval in seconds.

**Output interval for diagnostics:** 21600

**Write ensemble members:** specifies if a text output of ensemble members is to be written. Acceptable values are:

Value	Description
0	Do not write ensemble members
1	Write ensemble members

**Write ensemble members:** 0

**Write observations:** specifies if a text output of the observations is to be written. Acceptable values are:

Value	Description
0	Do not write observations
1	Write observations

**Write observations:** 0

**Write innovations:** specifies if a text output of the normalized innovations is to be written. Acceptable values are:

Value	Description
0	Do not write innovations
1	Write innovations

**Write innovations:** 0

Perturbation options

**Forcing perturbation algorithm:** specifies the algorithm for perturbing the forcing variables

Acceptable values are:

Value	Description
0	None
1	Static random gaussian perturbations for all forcing variables except precip and Static random lognormal perturbations for precip
2	GMAO perturbation algorithm

**Forcing perturbation algorithm:** 0

**Observation perturbation algorithm:** specifies the algorithm for perturbing the observations. Acceptable values are:

Value	Description
0	None
1	Static random gaussian perturbations
2	GMAO perturbation algorithm

**Observation perturbation algorithm:** 0

**State perturbation algorithm:** specifies the algorithm for perturbing the state prognostic variables Acceptable values are:

Value	Description
0	None
1	Static random gaussian perturbations
2	GMAO perturbation algorithm

**State perturbation algorithm:** 0

**Forcing perturbation frequency:** specifies the forcing perturbation interval in seconds

**Forcing perturbation frequency:** 3600

**Observation perturbation frequency:** specifies the observation perturbation interval in seconds

**Observation perturbation frequency:** 3600

**State perturbation frequency:** specifies the prognostic variable perturbation interval in seconds

**State perturbation frequency:** 3600

**Number of forcing fields to be perturbed:** Specifies the number of forcing fields to be perturbed.

**Number of forcing fields to be perturbed:**

**Forcing attributes file:** ASCII file that specifies the attributes of the forcing.

**Forcing attributes file:**

**Forcing perturbation attributes file:** ASCII file that specifies the attributes of the forcing perturbations.

**Forcing perturbation attributes file:**

**State attributes file:** ASCII file that specifies the attributes of the prognostic variables.

**State attributes file:**

**State perturbation attributes file:** ASCII file that specifies the attributes of the prognostic variable perturbations.

**State perturbation attributes file:**

**Observation attributes file:** ASCII file that specifies the attributes of the observation variables.

**Observation attributes file:**

**Observation perturbation attributes file:** ASCII file that specifies the attributes of the observation variable perturbations.

**Observation perturbation attributes file:**

**Synthetic SCA Assimilation:**

**Synthetic SCA directory:** specifies the directory for the synthetic snow covered area data.

**Synthetic SCA directory:**

**Synthetic Soil Moisture Assimilation**

**Synthetic Soil Moisture data directory:** specifies the directory for the synthetic soil moisture data.

**Synthetic Soil Moisture data directory:** ./input/dainput/SynSM/

#### Synthetic SWE Assimilation

**Synthetic SWE data directory:** specifies the directory for the synthetic snow water equivalent data.

**Synthetic SWE data directory:**                   `./input/dainput/SynSWE/`

#### AMSR-E smc Assimilation

**AMSR-E Soil Moisture data directory:** specifies the directory for the AMSR-E soil moisture data.

**AMSR-E Soil Moisture data directory:**       `'input'`

**AMSR-E and Noah CDF file directory:** specifies the directory for the AMSR-E and Noah CDF file.

**AMSR-E and Noah CDF file directory:**       `'input'`

**CDF levels:** specifies the number of CDF levels.

**CDF levels:** 20

#### SNODEP Assimilation

**SNODEP data directory:** specifies the directory for the SNODEP data.

**SNODEP data directory:**                   `./FORCING/AFWA1`

#### ISSCP Tskin Assimilation

**ISSCP Tskin data directory:** specifies the directory for the International Satellite Cloud Climatology Project (ISSCP) Tskin data.

**ISSCP Tskin data directory:**

### Synthetic LST Assimilation

**Synthetic LST data directory:** specifies the directory for the synthetic land surface temperature data.

**Synthetic LST data directory:**

### PRISM

**PRISM climatology filename:** specifies the file containing PRISM climatology data.

**PRISM climatology filename:**

**PRISM climatology x-dimension size:** specifies the number of columns in the PRISM climatology data.

**PRISM climatology x-dimension size:**

**PRISM climatology y-dimension size:** specifies the number of rows in the PRISM climatology data.

**PRISM climatology y-dimension size:**

### Bias estimation

**Bias estimation algorithm:** specifies the bias estimation algorithm to use.

**Bias estimation algorithm:**

**Bias estimation attributes file:** ASCII file that specifies the attributes of the bias estimation.

**Bias estimation attributes file:**

## 8.5 Domain specification

This section of the config file specifies the running domain (domain over which the simulation is carried out). The specification of the running domain section depends on the type of LIS domain and projection used. Section 8.1 lists the projections that LIS supports.

```
#Definition of Running Domain  
#Specify the domain extremes in latitude and longitude
```

### 8.5.1 Cylindrical lat/lon

This section describes how to specify a cylindrical latitude/longitude projection. See Appendix A for more details about setting these values.

```
run domain lower left lat:          25.875  
run domain lower left lon:         -124.875  
run domain upper right lat:        52.875  
run domain upper right lon:       -67.875  
run domain resolution dx:          0.25  
run domain resolution dy:          0.25
```

### 8.5.2 Polar stereographic

This section describes how to specify a polar stereographic projection. See Appendix B for more details about setting these values.

```
run domain lower left lat:          32.875  
run domain lower left lon:         -104.875  
run domain true lat:              36.875  
run domain standard lon:          -98.875  
run domain orientation:           0.0  
run domain resolution:            25.0  
run domain x-dimension size:      40  
run domain y-dimension size:      30
```

### 8.5.3 Lambert conformal

This section describes how to specify a Lambert conformal projection. See Appendix D for more details about setting these values.

```
run domain lower left lat:           32.875
run domain lower left lon:          -104.875
run domain true lat1:                36.875
run domain true lat2:                36.875
run domain standard lon:            -98.875
run domain resolution:               25.0
run domain x-dimension size:        40
run domain y-dimension size:        30
```

#### 8.5.4 Mercator

This section describes how to specify a Mercator projection. See Appendix E for more details about setting these values.

```
run domain lower left lat:           32.875
run domain lower left lon:          -104.875
run domain true lat1:                36.875
run domain standard lon:            -98.875
run domain resolution:               25.0
run domain x-dimension size:        40
run domain y-dimension size:        30
```

#### 8.5.5 Gaussian

This section describes how to specify a Gaussian projection. See Appendix C for more details about setting these values.

```
run domain first grid point lat:     -89.27665
run domain first grid point lon:      0.0
run domain last grid point lat:      89.27665
run domain last grid point lon:     -0.9375
run domain resolution dlon:          0.9375
run domain number of lat circles:    95
```

#### 8.5.6 Polar global

This section describes how to specify a polar global projection. See Appendix F for more details about setting these values.

This projection is not supported!

```

run domain number of columns:
run domain number of rows:
run domain origin lat of NH:
run domain origin lon of NH:
run domain origin lat of SH:
run domain origin lon of SH:
run domain orientation of NH:
run domain orientation of SH:
run domain mesh size:

#Definition of Parameter Domain

```

### 8.5.7 Cylindrical lat/lon

This section describes how to specify a cylindrical latitude/longitude projection. See Appendix A for more details about setting these values.

param domain lower left lat:	-59.875
param domain lower left lon:	-179.875
param domain upper right lat:	89.875
param domain upper right lon:	179.875
param domain resolution dx:	0.25
param domain resolution dy:	0.25

### 8.5.8 Gaussian

This section describes how to specify a Gaussian projection. See Appendix C for more details about setting these values.

param domain first grid point lat:	-89.27665
param domain first grid point lon:	0.0
param domain last grid point lat:	89.27665
param domain last grid point lon:	-0.9375
param domain resolution dlon:	0.9375
param domain number of lat circles:	95

### 8.5.9 Polar stereographic

This section describes how to specify a polar stereographic projection. See Appendix B for more details about setting these values.

```
param domain lower left lat:  
param domain lower left lon:  
param domain true lat:  
param domain standard lon:  
param domain orientation:  
param domain resolution:  
param domain x-dimension size:  
param domain y-dimension size:
```

## 8.6 Parameters

```
#Metadata for Parameter maps  
#Landcover and Landmask
```

**landmask file:** specifies the location of land/water mask file.

```
landmask file:          ./input/UMD-25KM/UMD_mask0.25.1gd4r
```

**landcover file:** specifies the location of the vegetation classification

```
landcover file:          ./input/UMD-25KM/UMD_veg0.25.1gd4r
```

See Appendix A for more details about setting these values.

```
landcover lower left lat:      -59.875  
landcover lower left lon:      -179.875  
landcover upper right lat:     89.875  
landcover upper right lon:     179.875  
landcover resolution (dx):    0.25  
landcover resolution (dy):    0.25
```

See Appendix C for more details about setting these values.

```
landcover first grid point lat:  
landcover first grid point lon:  
landcover last grid point lat:  
landcover last grid point lon:  
landcover resolution dlon:  
landcover number of lat circles:
```

See Appendix B for more details about setting these values.

```
landcover lower left lat:  
landcover lower left lon:  
landcover true lat:  
landcover standard lon:  
landcover orientation:  
landcover resolution:  
landcover x-dimension size:  
landcover y-dimension size:
```

Topography maps **elevation map**: specifies the elevation of the LIS grid.

```
elevation map: ./input/UMD-25KM/lis_elev.1gd4r
```

**slope map**: specifies the slope of the LIS grid.

```
slope map:
```

**aspect map**: specifies the aspect of the LIS grid.

```
aspect map:
```

**curvature map**: specifies the curvature of the LIS grid.

**curvature map:**

See Appendix A for more details about setting these values.

```
topography lower left lat:      -59.875
topography lower left lon:      -179.875
topography upper right lat:     89.875
topography upper right lon:    179.875
topography resolution (dx):    0.25
topography resolution (dy):    0.25
```

See Appendix C for more details about setting these values.

```
topography domain first grid point lat:      -89.27665
topography domain first grid point lon:        0.0
topography domain last grid point lat:       89.27665
topography domain last grid point lon:      -0.9375
topography domain resolution dlon:        0.9375
topography domain number of lat circles:   95
```

Soils maps **soil texture map**: specifies the soil texture file.

**soil texture map:**

**sand fraction map**: specifies the sand fraction map file.

```
sand fraction map:           ./input/UMD-25KM/sandfao.1gd4r
```

**clay fraction map**: specifies the clay fraction map file.

```
clay fraction map:           ./input/UMD-25KM/clayfao.1gd4r
```

**silt fraction map:** specifies the silt map file.

**silt fraction map:**           ./input/UMD-25KM/siltfao.1gd4r

**soil color map:** specifies the soil color map file.

**soil color map:**

**porosity map:** specifies porosity.

**porosity map:**

**saturated matric potential map:** specifies saturated matric potential

**saturated matric potential map:**           # psisat

**saturated hydraulic conductivity map:** specifies saturated hydraulic conductivity

**saturated hydraulic conductivity map:** # ksat

**b parameter map:** specifies b parameter

**b parameter map:**

**quartz map:** specifies quartz data

**quartz map:**

See Appendix A for more details about setting these values.

```
soils lower left lat:          -59.875
soils lower left lon:          -179.875
soils upper right lat:         89.875
soils upper right lon:         179.875
soils resolution (dx):        0.25
soils resolution (dy):        0.25
```

See Appendix C for more details about setting these values.

```
soils domain first grid point lat:      -89.27665
soils domain first grid point lon:       0.0
soils domain last grid point lat:       89.27665
soils domain last grid point lon:      -0.9375
soils domain resolution dlon:          0.9375
soils domain number of lat circles:    95
```

Albedo maps **albedo map**: specifies the source of the climatology based albedo files. The climatology albedo data files have the following naming convention:  
[directory]/[file header].[tag].1gd4r The tag should be either sum, win, spr, or aut depending on the season. The file header can be anything (such as alb1KM).

```
albedo map:                  ./input/UMD-25KM/alb
```

**albedo climatology interval**: specifies the frequency of the albedo climatology in months.

```
albedo climatology interval: 3
```

**max snow free albedo map**: specifies the static upper bound of snow free albedo file.

```
max snow free albedo map: ./input/UMD-25KM/global_mxsnoalb.25km.1gd4r
```

**bottom temperature map:** specifies the bottom boundary temperature data.

**bottom temperature map:**      . /input/UMD-25KM/tbot\_0.25.1gd4r

Greenness fraction maps **greenness fraction map:** specifies the source of the climatology based gfrac files. The climatology greenness data files have the following naming convention: `directory/file header.tag.1gd4r` The tag should represent the month (such as jan, feb, mar, etc. ). The file header can be anything (such as green1KM).

**greenness fraction map:**                            . /input/UMD-25KM/green

**greenness climatology interval:** specifies the frequency of the greenness climatology in months.

**greenness climatology interval:**                    1

LAI/SAI maps **LAI map:** specifies the source of the LAI climatology files. The climatology LAI data files have the following naming convention: `directory/file header.tag.1gd4r` The tag should represent the month (such as jan, feb, mar, etc. ). The file header can be anything (such as LAI1KM).

**LAI map:**    . /input/UMD-25KM/AVHRR\_LAI\_CLIM

**SAI map:** specifies the source of the SAI climatology files. The climatology SAI data files have the following naming convention: `directory/file header.tag.1gd4r` The tag should represent the month (such as jan, feb, mar, etc. ). The file header can be anything (such as SAI1KM).

**SAI map:**    . /input/UMD-25KM/AVHRR\_SAI\_CLIM

snow depth maps **Snow depth map:** specifies the location of the realtime snow depth.

**Snow depth map:**                                    . /FORCING/AFWA1

## 8.7 Forcings

### 8.7.1 GDAS

**GDAS forcing directory:** specifies the location of the GDAS forcing files.

GDAS forcing directory:                   `./input/FORCING/GDAS/`

**GDAS T126 elevation map:** specifies the GDAS T126 elevation definition.

GDAS T126 elevation map:

**GDAS T170 elevation map:** specifies the GDAS T170 elevation definition.

GDAS T170 elevation map:                   `./input/UMD-25KM/gdas_T170_elev.1gd4r`

**GDAS T254 elevation map:** specifies the GDAS T254 elevation definition.

GDAS T254 elevation map:                   `./input/UMD-25KM/gdas_T254_elev.1gd4r`

**GDAS T382 elevation map:** specifies the GDAS T382 elevation definition.

GDAS T382 elevation map:                   `./input/UMD-25KM/gdas_T382_elev.1gd4r`

**GDAS domain x-dimension size:** specifies the number of columns of the native domain parameters of the GDAS forcing data. The map projection is specified in the driver modules defined for the GDAS routines.

**GDAS domain x-dimension size:** 512

**GDAS domain y-dimension size:** specifies the number of rows of the native domain parameters of the GDAS forcing data. The map projection is specified in the driver modules defined for the GDAS routines.

**GDAS domain y-dimension size:** 256

**GDAS number of forcing variables:** specifies the number of forcing variables provided by GDAS at the model initialization step.

**GDAS number of forcing variables:** 10

### 8.7.2 GEOS

**GEOS forcing directory:** specifies the location of the GEOS forcing files.

**GEOS forcing directory:** ./input/FORCING/GEOS/BEST\_LK/

**GEOS domain x-dimension size:** specifies the number of columns of the native domain parameters of the GEOS forcing data. The map projection is specified in the driver modules defined for the GEOS routines.

**GEOS domain x-dimension size:** 360

**GEOS domain y-dimension size:** specifies the number of rows of the native domain parameters of the GEOS forcing data. The map projection is specified in the driver modules defined for the GEOS routines.

**GEOS domain y-dimension size:** 181

**GEOS number of forcing variables:** specifies the number of forcing variables provided by GEOS at the model initialization step.

**GEOS number of forcing variables:** 13

### 8.7.3 ECMWF

**ECMWF forcing directory:** specifies the location of the ECMWF forcing files.

**ECMWF forcing directory:** ./input/FORCING/ECMWF/

**ECMWF elevation map 0:** specifies the ECMWF elevation definition.

**ECMWF elevation map 0:**

**ECMWF elevation map 1:** specifies the ECMWF elevation definition.

**ECMWF elevation map 1:**

**ECMWF elevation map 2:** specifies the ECMWF elevation definition.

**ECMWF elevation map 2:**

**ECMWF domain x-dimension size:** specifies the number of columns of the native domain parameters of the ECMWF forcing data. The map projection is specified in the driver modules defined for the ECMWF routines.

**ECMWF domain x-dimension size:**

**ECMWF domain y-dimension size:** specifies the number of rows of the native domain parameters of the ECMWF forcing data. The map projection is specified in the driver modules defined for the ECMWF routines.

**ECMWF domain y-dimension size:**

**ECMWF number of forcing variables:** specifies the number of forcing variables provided by ECMWF at the model initialization step.

**ECMWF number of forcing variables:**

#### 8.7.4 NLDAS

**NLDAS forcing directory:** specifies the location of the NLDAS forcing files.

**NLDAS forcing directory:**                           **./input/FORCING/NLDAS**

**NLDAS domain x-dimension size:** specifies the number of columns of the native domain parameters of the NLDAS forcing data. The map projection is specified in the driver modules defined for the NLDAS routines.

**NLDAS domain x-dimension size:**                   **464**

**NLDAS domain y-dimension size:** specifies the number of rows of the native domain parameters of the NLDAS forcing data. The map projection is specified in the driver modules defined for the NLDAS routines.

**NLDAS domain y-dimension size:**                   **224**

**NLDAS elevation map:** specifies the NLDAS elevation definition.

NLDAS elevation difference map: ./input/FORCING/NLDAS/elevdiff.bin

EDAS height map: specifies the EDAS height definition.

EDAS height map: ./input/N0.125/edas\_elev.1gd4r

EDAS height lower left lat: specifies the lower left latitude of the EDAS domain. (cylindrical latitude/longitude projection)

EDAS height lower left lat: 25.0625

EDAS height lower left lat: specifies the lower left longitude of the EDAS domain. (cylindrical latitude/longitude projection)

EDAS height lower left lon: -124.9375

EDAS height lower left lat: specifies the upper right latitude of the EDAS domain. (cylindrical latitude/longitude projection)

EDAS height upper right lat: 52.9375

EDAS height lower left lat: specifies the upper right longitude of the EDAS domain. (cylindrical latitude/longitude projection)

EDAS height upper right lon: -67.0625

EDAS height lower left lat: specifies the longitudinal resolution of the EDAS domain. (cylindrical latitude/longitude projection)

EDAS height resolution (dx): 0.125

**EDAS height lower left lat:** specifies the latitudinal resolution of the EDAS domain. (cylindrical latitude/longitude projection)

EDAS height resolution (dy): 0.125

### 8.7.5 GSWP

**GSWP landmask file:** specifies the GSWP2 landmask file.

GSWP landmask file: ./input/gswp2data/Fixed/landmask\_gswp.nc

**GSWP domain x-dimension size:** specifies the number of columns of the GSWP2 domain.

GSWP domain x-dimension size: 360

**GSWP domain y-dimension size:** specifies the number of rows of the GSWP2 domain.

GSWP domain y-dimension size: 150

**GSWP number of forcing variables:** specifies the number of GSWP2 forcing variables.

GSWP number of forcing variables: 10

**GSWP 2m air temperature map:** specifies the GSWP2 2 meter air temperature data.

GSWP 2m air temperature map: ./input/gswp2data/Tair\_cru/Tair\_cru

**GSWP 2m specific humidity map:** specifies the GSWP2 2 meter specific humidity data.

**GSWP 2m specific humidity map:**                   ./input/gswp2data/Qair\_cru/Qair\_cru

**GSWP wind map:** specifies the GSWP2 wind data.

**GSWP wind map:**                                   ./input/gswp2data/Wind\_ncep/Wind\_ncep

**GSWP surface pressure map:** specifies the GSWP2 surface pressure data.

**GSWP surface pressure map:**                   ./input/gswp2data/PSurf\_ecor/PSurf\_ecor

**GSWP convective rainfall rate map:** specifies the GSWP2 convective rainfall rate data.

**GSWP convective rainfall rate map:**           ./input/gswp2data/Rainf\_C\_gswp/Rainf\_C\_gswp

**GSWP rainfall rate map:** specifies the GSWP2 rainfall rate data.

**GSWP rainfall rate map:**                           ./input/gswp2data/Rainf\_gswp/Rainf\_gswp

**GSWP snowfall rate map:** specifies the GSWP2 snowfall rate data.

**GSWP snowfall rate map:**                           ./input/gswp2data/Snowf\_gswp/Snowf\_gswp

**GSWP incident shortwave radiation map:** specifies the GSWP2 incident shortwave radiation data.

**GSWP incident shortwave radiation map:** ./input/gswp2data/SWdown\_srb/SWdown\_srb

**GSWP incident longwave radiation map:** specifies the GSWP2 incident longwave radiation data.

```
GSWP incident longwave radiation map: ./input/gswp2data/LWdown_srb/LWdown_srb
```

#### 8.7.6 ECMWF Reanalysis

**ECMWF Reanalysis forcing directory:** specifies the location of the ECMWF Reanalysis forcing files.

```
ECMWF Reanalysis forcing directory: ./input/FORCING/ECMWF-REANALYSIS/
```

**ECMWF Reanalysis maskfile:** specifies the file containing the ECMWF Reanalysis land/sea mask.

```
ECMWF Reanalysis maskfile: ./input/FORCING/ECMWF-REANALYSIS/ecmwf_land_sea.05
```

**ECMWF Reanalysis elevation map:** specifies the file containing the ECMWF Reanalysis elevation definition.

```
ECMWF Reanalysis elevation map: ./input/UMD-25KM/ecmwfreanal_elev.1gd4r
```

**ECMWF Reanalysis domain x-dimension size:** specifies the number of columns of the ECMWF Reanalysis domain.

```
ECMWF Reanalysis domain x-dimension size: 720
```

**ECMWF Reanalysis domain y-dimension size:** specifies the number of rows of the ECMWF Reanalysis domain.

```
ECMWF Reanalysis domain y-dimension size: 360
```

### **8.7.7 AFWA/AGRMET**

**AGRMET forcing directory:** specifies the location of the root directory containing the input files. The AGRMET processing algorithms assumes the following hierarchy under the root directory at each instance. For example, if the root directory for storing the files is 'FORCING/AFWA/', and the current instance is decemeber 1st, 2005, then the files are stored under the 'FORCING/AFWA/20051201/' directory. The additional directory names specified here are assumed to be located under this root directory.

AGRMET forcing directory:                   ./FORCING/

**AGRMET analysis directory:** specifies the location where temporary precip analysis fields will be written

AGRMET analysis directory:                   ./Analysis

**AGRMET surface fields directory:** specifies the location of the surface fields (sfc\*).

AGRMET surface fields directory:           SFCALC

**AGRMET merged precip directory:** specifies the location of the processed precip obs (presav\_\*).

AGRMET merged precip directory:           PRECIP

**AGRMET merged precip directory:** specifies the location of the WWMCA data (WWMCA\*).

AGRMET cloud data directory:           WWMCA

**AGRMET GFS data directory:** specifies the location of the GFS data (**MT.avn\***).

**AGRMET GFS data directory:** GFS

**AGRMET SSMI data directory:** specifies the location of the SSM/I data (**ssmira\_\***).

**AGRMET SSMI data directory:** SSMI

**AGRMET GEOPRECIP data directory:** specifies the location of the GEOPRECIP files (**prec08\*** and **rank08\***).

**AGRMET GEOPRECIP data directory:** GEO

**AGRMET CDMS data directory:** specifies the location of the surface and precip obs (**sfcobs\_\*** and **preobs\_\***).

**AGRMET CDMS data directory:** CDMS

**AGRMET use timestamp on directories:** specifies whether or not to use a timestamp on directories. Acceptable values are:

Value	Description
0	do not use timestamp
1	use timestamp

**AGRMET use timestamp on directories:** 1

**AGRMET polar mask file:** specifies the landmask in 8th mesh polar stereographic projection used by the AGRMET algorithms.

**AGRMET polar mask file:** ./STATIC/point\_switches

**AGRMET polar terrain file:** specifies the terrain in 8th mesh polar stereographic projection used by the AGRMET algorithms.

AGRMET polar terrain file: . /STATIC/terrain

**AGRMET sfcalc cntm file:** specifies the name of the files with the spreading radii used for the barnes analysis on the GFS and surface obs.

AGRMET sfcalc cntm file: . /STATIC/sfcalc-cntm

**AGRMET precip climatology:** specifies the path to the precip climatology data.

AGRMET precip climatology: . /STATIC/pcp\_clim/

**AGRMET nogaps wind weight:** specifies the weighting factor for the first guess winds

AGRMET nogaps wind weight: 0.75

**AGRMET minimum wind speed:** specifies the minimum allowable wind speed on the AGRMET grid

AGRMET minimum wind speed: 0.25

**AGRMET use present/past weather estimate:** specifies whether to use present/past weather estimates. Acceptable values are:

Value	Description
0	do not use estimates
1	use estimates

AGRMET use present/past weather estimate: 1

**AGRMET use precip observations:** specifies whether to use precip observations. Acceptable values are:

Value	Description
0	do not use observations
1	use observations

AGRMET use precip observations: 1

**AGRMET use SSMI data:** specifies whether to use SSM/I data Acceptable values are:

Value	Description
0	do not use SSM/I data
1	use SSM/I data

AGRMET use SSMI data: 1

**AGRMET use CDFSII-based estimate:** specifies whether to use a CDFS-II based estimate. Acceptable values are:

Value	Description
0	do not use estimate
1	use estimate

AGRMET use CDFSII-based estimate: 1

**AGRMET use GEOPRECIP estimate:** specifies whether to use a GEOPRECIP based estimate. Acceptable values are:

Value	Description
0	do not use
1	use the estimate, do not use the rank
2	use the estimate and use the rank

AGRMET use GEOPRECIP estimate: 2

**AGRMET CDFSII time interval:** specifies the CDFS-II time interval to look for cloud amount. Current value is 6.

AGRMET CDFSII time interval: 6

AGRMET use precip climatology: specifies whether to use precip climatology.  
Acceptable values are:

Value	Description
0	do not use precip climatology
1	use precip climatology

AGRMET use precip climatology: 1

AGRMET SSMI zero use switch: specifies whether to use SSM/I zeros. Acceptable values are:

Value	Description
0	do not use zeros
1	use zeros

AGRMET SSMI zero use switch: 1

AGRMET snow distribution shape parameter: specifies the snow distribution shape parameter (A typical value is 2.6)

AGRMET snow distribution shape parameter: 2.6

AGRMET alternate monthly weighting factor: specifies the alternate monthly weighting factor used in the precip processing.

AGRMET alternate monthly weighting factor: 1.0

AGRMET minimum 3hr climo value: specifies a minimum 3 hour precip climo value required to generate a non-zero CDFSII total cloud-based precip estimate. A typical value is 0.025.

AGRMET minimum 3hr climo value: 0.025

AGRMET maximum 3hr climo value: specifies a maximum 3 hour precip climo value required to generate a non-zero CDFSII total cloud-based precip estimate. A typical value is 0.375.

AGRMET maximum 3hr climo value: 0.375

AGRMET minimum precip-per-precip day multiplier: specifies a minimum precip-per-precip day multiplier used to generate a non-zero CDFSII total cloud based precip estimate. A typical value is 0.0.

AGRMET minimum precip-per-precip day multiplier: 0.0

AGRMET maximum precip-per-precip day multiplier: specifies a maximum precip-per-precip day multiplier used to generate a non-zero CDFSII total cloud based precip estimate. A typical value is 1.1.

AGRMET maximum precip-per-precip day multiplier: 1.1

AGRMET cloud threshold to generate CDFSII estimate: specifies the cloud threshold to generate a CDFSII-based estimate. A typical value is 85.0.

AGRMET cloud threshold to generate CDFSII estimate: 85.0

AGRMET median cloud cover percentage1: specifies the median cloud cover percentage to move to for the CDFSII based precip estimate. A typical value is 15.0.

AGRMET median cloud cover percentage1: 15.0

AGRMET median cloud cover percentage2: specifies the median cloud cover percentage to move to for the CDFSII based precip estimate. A typical value is 0.60.

```
AGRMET median cloud cover percentage2: 0.60
```

**AGRMET overcast percentage:** specifies the overcast percentage to move to for CDFSII based precipitation estimate. A typical value is 0.30.

```
AGRMET overcast percentage: 0.30
```

**AGRMET 3hr maximum precip ceiling:** specifies the 3 hour maximum precip ceiling value. A typical value is 200.0.

```
AGRMET 3hr maximum precip ceiling: 200.0
```

### 8.7.8 NLDAS2

**NLDAS2 forcing directory:** specifies the location of the NLDAS2 forcing files.

```
NLDAS2 forcing directory: ./input/FORCING/NLDAS2
```

**NLDAS2 elevation map:** specifies the NLDAS2 elevation definition.

```
NLDAS2 elevation map:
```

**NLDAS2 domain x-dimension size:** specifies the number of columns of the native domain parameters of the NLDAS2 forcing data. The map projection is specified in the driver modules defined for the NLDAS2 routines.

```
NLDAS2 domain x-dimension size: 464
```

**NLDAS2 domain y-dimension size:** specifies the number of rows of the native domain parameters of the NLDAS2 forcing data. The map projection is specified in the driver modules defined for the NLDAS2 routines.

NLDAS2 domain y-dimension size: 224

### 8.7.9 PRINCETON

**PRINCETON forcing directory:** specifies the location of the PRINCETON forcing files.

PRINCETON forcing directory: ./input/FORCING/PRINCETON

**PRINCETON elevation map:** specifies the PRINCETON elevation definition.

PRINCETON elevation map: ./input/UMD-100KM/hydro1k\_elev\_mean\_1d.1gd4r

**PRINCETON domain x-dimension size:** specifies the number of columns of the native domain parameters of the PRINCETON forcing data. The map projection is specified in the driver modules defined for the PRINCETON routines.

PRINCETON domain x-dimension size: 360

**PRINCETON domain y-dimension size:** specifies the number of rows of the native domain parameters of the PRINCETON forcing data. The map projection is specified in the driver modules defined for the PRINCETON routines.

PRINCETON domain y-dimension size: 180

**PRINCETON number of forcing variables:** specifies the number of forcing variables provided by PRINCETON at the model initialization step.

```
PRINCETON number of forcing variables: 7
```

### 8.7.10 SALDAS

SALDAS forcing directory: specifies the location of the SALDAS forcing files.

```
SALDAS forcing directory:          ./input/FORCING/SALDAS
```

SALDAS elevation map: specifies the SALDAS elevation definition.

```
SALDAS elevation map:
```

SALDAS elevation map: specifies the SALDAS elevation change 1 definition.

```
SALDAS elevation map change 1:
```

SALDAS elevation map: specifies the SALDAS elevation change 2 definition.

```
SALDAS elevation map change 2:
```

SALDAS elevation map: specifies the SALDAS elevation change 3 definition.

```
SALDAS elevation map change 3:
```

SALDAS domain x-dimension size: specifies the number of columns of the native domain parameters of the SALDAS forcing data. The map projection is specified in the driver modules defined for the SALDAS routines.

**SALDAS domain x-dimension size:**

**SALDAS domain y-dimension size:** specifies the number of rows of the native domain parameters of the SALDAS forcing data. The map projection is specified in the driver modules defined for the SALDAS routines.

**SALDAS domain y-dimension size:**

**SALDAS number of forcing variables:** specifies the number of forcing variables provided by SALDAS at the model initialization step.

**SALDAS number of forcing variables:**

### 8.7.11 Rhone AGG

**Rhone AGG forcing directory:** specifies the location of the Rhone AGG forcing files.

**Rhone AGG forcing directory:**           `./input/FORCING/RHONE`

**Rhone AGG domain x-dimension size:** specifies the number of columns of the native domain parameters of the Rhone AGG forcing data. The map projection is specified in the driver modules defined for the Rhone AGG routines.

**Rhone AGG domain x-dimension size:** 5

**Rhone AGG domain y-dimension size:** specifies the number of rows of the native domain parameters of the Rhone AGG forcing data. The map projection is specified in the driver modules defined for the Rhone AGG routines.

**Rhone AGG domain y-dimension size:** 6

### 8.7.12 GSWP1

**GSWP1 forcing directory:** specifies the location of the GSWP1 forcing files.

```
GSWP1 forcing directory:      ./input/FORCING/GSWP1
```

**GSWP1 domain x-dimension size:** specifies the number of columns of the native domain parameters of the GSWP1 forcing data. The map projection is specified in the driver modules defined for the GSWP1 routines.

```
GSWP1 domain x-dimension size: 360
```

**GSWP1 domain y-dimension size:** specifies the number of rows of the native domain parameters of the GSWP1 forcing data. The map projection is specified in the driver modules defined for the GSWP1 routines.

```
GSWP1 domain y-dimension size: 150
```

**GSWP1 number of forcing variables:** specifies the number of forcing variables provided by GSWP1 at the model initialization step.

```
GSWP1 number of forcing variables: 9
```

## 8.8 Supplemental forcings

### 8.8.1 AGRMET radiation

**AGRRAD forcing directory:** specifies the directory containing AGRMET radiation data.

**AGRRAD forcing directory:**                   `./input/FORCING/AGRRAD`

### **8.8.2 CMAP precipitation**

**CMAP forcing directory:** specifies the location of the CMAP forcing files.

**CMAP forcing directory:**                   `./input/FORCING/CMAP`

**CMAP domain x-dimension size:** specifies the number of columns of the native domain parameters of the CMAP forcing data. The map projection is specified in the driver modules defined for the CMAP routines.

**CMAP domain x-dimension size:**           512

**CMAP domain y-dimension size:** specifies the number of rows of the native domain parameters of the CMAP forcing data. The map projection is specified in the driver modules defined for the CMAP routines.

**CMAP domain y-dimension size:**           256

### **8.8.3 CEOP station data**

CEOP station forcing – during EOP1

**CEOP location index:** specifies the location of the CEOP station.

**CEOP location index:**                   3 #SGP location

**CEOP forcing directory:** specifies the location of the CEOP forcing files.

**CEOP forcing directory:**                   `./input/FORCING/CEOP/sgp.cfr`

**CEOP metadata file:** specifies the file containing CEOP metadata.

**CEOP metadata file:**                   `./input/FORCING/CEOP/sgp.mdata`

#### **8.8.4 SCAN station data**

**SCAN forcing directory:** specifies the location of the SCAN forcing files.

**SCAN forcing directory:**                   `./input/FORCING/SCAN`

**SCAN metadata file:** specifies the file containing SCAN metadata.

**SCAN metadata file:**                   `./input/FORCING/SCAN/msu_scan.mdata`

#### **8.8.5 TRMM 3B42RT precipitation**

**TRMM 3B42RT forcing directory:** specifies the location of the TRMM 3B42RT forcing files.

**TRMM 3B42RT forcing directory:**           `./input/FORCING/3B42RT/`

**TRMM 3B42RT domain x-dimension size:** specifies the number of columns of the native domain parameters of the TRMM 3B42RT forcing data. The map projection is specified in the driver modules defined for the TRMM 3B42RT routines.

```
TRMM 3B42RT domain x-dimension size: 1440
```

**TRMM 3B42RT domain y-dimension size:** specifies the number of rows of the native domain parameters of the TRMM 3B42RT forcing data. The map projection is specified in the driver modules defined for the TRMM 3B42RT routines.

```
TRMM 3B42RT domain y-dimension size: 480
```

#### 8.8.6 TRMM 3B42V6 precipitation

**TRMM 3B42V6 forcing directory:** specifies the location of the TRMM 3B42V6 forcing files.

```
TRMM 3B42V6 forcing directory:      ./input/FORCING/3B42V6/
```

**TRMM 3B42V6 domain x-dimension size:** specifies the number of columns of the native domain parameters of the TRMM 3B42V6 forcing data. The map projection is specified in the driver modules defined for the TRMM 3B42V6 routines.

```
TRMM 3B42V6 domain x-dimension size: 1440
```

**TRMM 3B42V6 domain y-dimension size:** specifies the number of rows of the native domain parameters of the TRMM 3B42V6 forcing data. The map projection is specified in the driver modules defined for the TRMM 3B42V6 routines.

```
TRMM 3B42V6 domain y-dimension size: 400
```

### **8.8.7 CMORPH precipitation**

**CMORPH forcing directory:** specifies the location of the CMORPH forcing files.

```
CMORPH forcing directory:      ./input/FORCING/CMORPH/
```

**CMORPH domain x-dimension size:** specifies the number of columns of the native domain parameters of the CMORPH forcing data. The map projection is specified in the driver modules defined for the CMORPH routines.

```
CMORPH domain x-dimension size: 4989
```

**CMORPH domain y-dimension size:** specifies the number of rows of the native domain parameters of the CMORPH forcing data. The map projection is specified in the driver modules defined for the CMORPH routines.

```
CMORPH domain y-dimension size: 1649
```

### **8.8.8 Stage II precipitation**

**STAGE2 forcing directory:** specifies the location of the STAGE2 forcing files.

```
STAGE2 forcing directory:          ./input/FORCING/STII
```

### **8.8.9 Stage IV precipitation**

**STAGE4 forcing directory:** specifies the location of the STAGE4 forcing files.

```
STAGE4 forcing directory:          ./input/FORCING/STIV
```

### 8.8.10 D2PCPCAR

D2PCPCAR forcing directory: specifies the location of the D2PCPCAR forcing files.

```
D2PCPCAR forcing directory:
```

### 8.8.11 D2PCPOKL

D2PCPOKL forcing directory: specifies the location of the D2PCPOKL forcing files.

```
D2PCPOKL forcing directory:
```

## 8.9 Land surface models

### 8.9.1 Forcing only – Template

TEMPLATE model output interval: defines the output interval for the template LSM, in seconds. The template LSM is not a model; rather, it is a placeholder for a model. It demonstrates the hooks that are needed to add a land surface model into LIS. This “LSM” is also used to run LIS with the purpose of only processing and writing forcing data.

```
TEMPLATE model output interval:      3600    #in seconds
```

### 8.9.2 NCEP's Noah

**NOAH model output interval:** defines the output interval for Noah, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

NOAH model output interval: 10800

**NOAH restart output interval:** defines the restart writing interval for Noah, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

NOAH restart output interval: 86400

**NOAH restart file:** specifies the Noah active restart file.

NOAH restart file:

**NOAH slope file:** specifies the Noah static slope file.

NOAH slope file:

**NOAH vegetation parameter table:** specifies the Noah static vegetation parameter table file.

NOAH vegetation parameter table: ./input/noah\_parms/noah.vegparms.txt

**NOAH soil parameter table:** specifies the Noah soil parameter file.

NOAH soil parameter table: ./input/noah\_parms/noah.soilparms.txt

**NOAH general parameter table:** specifies the Noah general parameter file.

**NOAH general parameter table:** ./input/noah\_parms/GENPARM.UNIF.TBL

**NOAH bottom temperature climatology interval:** specifies in months, the climatology interval of the TBOT files. 0 indicates that the files are static.

**NOAH bottom temperature climatology interval:** 0

**NOAH number of vegetation parameters:** specifies the number of static vegetation parameters specified for each veg type.

**NOAH number of vegetation parameters:** 7

**NOAH soils scheme:** specifies the soil mapping scheme used. Acceptable values are:

Value	Description
1	Zobler
2	STATSGO

**NOAH soils scheme:** 1

**NOAH number of soil classes:** specifies the number soil classes in the above mapping scheme Acceptable values are:

Value	Description
9	Zobler
19	STATSGO

**NOAH number of soil classes:** 9

**NOAH number of soil layers:** specifies the number of soil layers. The typical value used in Noah is 4.

**NOAH number of soil layers:** 4

**NOAH observation height:** specifies the height in meters of meteorological observations. The typical value used in Noah is 6 meters.

NOAH observation height: 20 #meters

**NOAH initial soil moisture:** specifies the initial volumetric soil moisture used in the cold start runs. (units  $\frac{m^3}{m^3}$ )

NOAH initial soil moisture: 0.20 #volumetric soil moisture (m3 m-3)

**NOAH initial soil temperature:** specifies the initial skin temperature in Kelvin used in the cold start runs.

NOAH initial soil temperature: 290.0 # Kelvin

### 8.9.3 CLM 2.0

**CLM model output interval:** defines the output interval for CLM, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

CLM model output interval: 10800

**CLM restart output interval:** defines the restart writing interval for CLM, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

CLM restart output interval: 86400

**CLM restart file:** specifies the CLM active restart file.

CLM restart file:

**CLM vegetation parameter file:** specifies vegetation type parameters look-up table.

CLM vegetation parameter table:           `./input/clm_parms/umdvegparam.txt`

**CLM canopy height table:** specifies the canopy top and bottom heights (for each vegetation type) look-up table.

CLM canopy height table:           `./input/clm_parms/clm2_ptcanhts.txt`

**CLM initial soil moisture:** specifies the initial volumetric soil moisture wetness used in the cold start runs.

CLM initial soil moisture:           0.45

**CLM initial soil temperature:** specifies the initial soil temperature in Kelvin used in the cold start runs.

CLM initial soil temperature:           290.0

**CLM initial snow mass:** specifies the initial snow mass used in the cold start runs.

CLM initial snow mass:           0.0

#### 8.9.4 VIC

**CLM model output interval:** defines the output interval for CLM, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

VIC model output interval:

**VIC restart output interval:** defines the restart writing interval for VIC, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

**VIC restart output interval:**

**VIC number of soil layers:** specifies the number of soil layers. The typical value used in VIC is 4.

**VIC number of soil layers:**

**VIC number of soil thermal nodes:** specifies the number of soil thermal nodes. The typical value used in VIC is 4.

**VIC number of soil thermal nodes:**

**VIC number of snow bands:** specifies the number of snow bands. The typical value used in VIC is 4.

**VIC number of snow bands:**

**VIC number of rootzones:** specifies the number of rootzones. The typical value used in VIC is 4.

**VIC number of rootzones:**

**VIC full energy balance mode:** specifies whether or not VIC will run in full energy balance mode. Acceptable values are:

Value	Description
0	do not run in full energy mode
1	run in full energy mode

**VIC full energy balance mode:**

**VIC frozen soil mode:** specifies whether or not VIC will run in frozen soil mode. Acceptable values are:

Value	Description
0	do not run in frozen soil mode
1	run in frozen soil mode

**VIC frozen soil mode:**

**VIC soil parameter table** specifies the soil parameters look-up table.

**VIC soil parameter table:**

**VIC vegetation parameter table** specifies the vegetation parameters look-up table.

**VIC vegetation parameter table:**

**VIC restart file:** specifies the VIC active restart file.

**VIC restart file:**

**VIC Ds map:** specifies the Ds map.

**VIC Ds map:**

**VIC DsMax map:** specifies the DsMax map.

**VIC DsMax map:**

**VIC Ws map:** specifies the Ws map.

VIC Ws map:

VIC infiltration capacity map: specifies the infiltration capacity map.

VIC infiltrataion capacity map:

VIC depth1 map: specifies the depth1 map.

VIC depth1 map:

VIC depth2 map: specifies the depth2 map.

VIC depth2 map:

VIC depth3 map: specifies the depth3 map.

VIC depth3 map:

VIC initial surf temp: specifies the initial surface temperature.

VIC initial surf temp:

### 8.9.5 Mosaic

Mosaic model output interval: defines the output interval for Mosaic, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

Mosaic model output interval: 10800

**Mosaic restart output interval:** defines the restart writing interval for Mosaic, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

Mosaic restart output interval: 86400

**Mosaic restart file:** specifies the Mosaic active restart file.

Mosaic restart file:

**Mosaic vegetation parameter table:** specifies the vegetation parameters look-up table.

Mosaic vegetation parameter table: ./input/mos\_parms/mosaic\_vegparms\_umd.txt

**Mosaic monthly vegetation parameter table:** specifies the monthly vegetation parameters look-up table.

Mosaic monthly vegetation parameter table: ./input/mos\_parms/mosaic\_monthlyvegparms\_umd.txt

**Mosaic soil parameter table:** specifies the soil parameters look-up table.

Mosaic soil parameter table: ./input/mos\_parms/mosaic\_soilparms\_fao.txt

**Mosaic number of soil classes:** specifies the number of soil classes. Acceptable values are:

Value	Description
11	FAO

Mosaic number of soil classes: 11

**Mosaic initial soil moisture:** specifies the initial soil moisture.

Mosaic initial soil moisture: 0.3

Mosaic initial soil temperature: specifies the initial soil temperature in Kelvin.

Mosaic initial soil temperature: 290

Mosaic Depth of Layer 1 (m): specifies the depth in meters of layer 1.

Mosaic Depth of Layer 1 (m): 0.02

Mosaic Depth of Layer 2 (m): specifies the depth in meters of layer 2.

Mosaic Depth of Layer 2 (m): 1.48

Mosaic Depth of Layer 3 (m): specifies the depth in meters of layer 3.

Mosaic Depth of Layer 3 (m): 2.00

#### 8.9.6 Hyssib

HYSSIB model output interval: defines the output interval for HYSSIB, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

HYSSIB model output interval: 10800

HYSSIB restart output interval: defines the restart writing interval for HYSSIB, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

**HYSSIB restart output interval:** 86400

**HYSSIB restart file:** specifies the HYSSIB active restart file.

**HYSSIB restart file:**

**HYSSIB vegetation parameter table:** specifies the HYSSIB static vegetation parameter table file.

**HYSSIB vegetation parameter table:** ./input/hyssib\_parms/hyssib\_vegparms.bin

**HYSSIB albedo parameter table:** specifies the HYSSIB static albedo parameter table file.

**HYSSIB albedo parameter table:** ./input/hyssib\_parms/hyssib\_albedo.bin

**HYSSIB topography stand dev file:** specifies the HYSSIB topography standard deviation file.

**HYSSIB topography stand dev file:** ./input/UMD-25KM/topo\_std.1gd4r

**HYSSIB bottom temperature climatology interval:** specifies the interval of the HYSSIB bottom temperature climatology.

**HYSSIB bottom temperature climatology interval:** 0

**HYSSIB number of vegetation parameters:** specifies the number of vegetation parameters.

**HYSSIB number of vegetation parameters:** 20

**HYSSIB number of vegetation parameters:** specifies the number of monthly vegetation parameters.

HYSSIB number of monthly veg parameters: 11

HYSSIB initial soil moisture: specifies the initial soil moisture.

HYSSIB initial soil moisture: 0.30

HYSSIB initial soil temperature: specifies the initial soil temperature in Kelvin.

HYSSIB initial soil temperature: 290.0

### 8.9.7 SiB2

**SiB2 model output interval:** defines the output interval for SiB2, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

SiB2 model output interval: 10800

**SiB2 restart output interval:** defines the restart writing interval for SiB2, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

SiB2 restart output interval: 86400

**SiB2 restart file:** specifies the SiB2 active restart file.

SiB2 restart file:

**SiB2 albedo and radiation parameter file:** specifies the SiB2 albedo and radiation parameter file.

```
SiB2 albedo and radiation parameter file: ./input/sib2_parms/ssibalb
```

**SiB2 monthly vegetation parameter file:** specifies the monthly vegetation parameter file.

```
SiB2 monthly vegetation parameter file: ./input/UMD-100KM/veg_month_1.0.1gd4r
```

**SiB2 static vegetation parameter file:** specifies the static vegetation parameter file.

```
SiB2 static vegetation parameter file: ./input/UMD-100KM/veg_const_1.0.1gd4r
```

**SiB2 initial soil moisture:** specifies the initial volumetric soil moisture.  
(units  $\frac{m^3}{m^3}$ )

```
SiB2 initial soil moisture: 0.20
```

**SiB2 initial soil temperature:** specifies the initial soil temperature in Kelvin.

```
SiB2 initial soil temperature: 290.0
```

### 8.9.8 Catchment

Catchment based tile map

**tile coord file:** specifies the tile coordinate file. This file maps the Catchment-based tiles to their overlying atmospheric grid.

```
tile coord file: ./input/cat_parms/PE_360x180_DE_288x270_DE_NO_TINY.rst
```

**tile veg file:** specifies the tile vegetation fraction file. This file contains the fractions of the various vegetation types for Catchment-based tile.

```
tile veg file:      ./input/cat_parms/mosaic_veg_typs_fracs
```

**Catchment model output interval:** defines the output interval for Catchment, in seconds. The typical value used in the LIS runs is 3 hours (=10800)

```
Catchment model output interval:      10800
```

**Catchment restart output interval:** defines the restart writing interval for Catchment, in seconds. The typical value used in the LIS runs is 24 hours (=86400).

```
Catchment restart output interval:    86400
```

**Catchment restart file:** specifies the Catchment active restart file.

```
Catchment restart file:
```

**Catchment roughness length table:** specifies the Catchment roughness length table.

```
Catchment roughness length table:
```

**Catchment veg height table:** specifies the Catchment vegetation height table.

```
Catchment veg height table:
```

**Catchment parameter directory:** specifies the directory containing the various Catchment parameter data.

**Catchment parameter directory:**           `./input/cat_parms/`

**Catchment MODIS directory:** specifies the directory containing the MODIS-based data.

**Catchment MODIS directory:**

**Catchment veg class file:** specifies the vegetation classification file.

**Catchment veg class file:**

**Catchment soil param file:** specifies the soil parameter file.

**Catchment soil param file:**

**Catchment surf layer ts file:** specifies the surface layer ts file.

**Catchment surf layer ts file:**

**Catchment topo ar file:** specifies the topography ar file.

**Catchment topo ar file:**

**Catchment topo bf file:** specifies the topography bf file.

**Catchment topo bf file:**

**Catchment topo ts file:** specifies the topography ts file.

**Catchment topo ts file:**

**Catchment initial soil moisture:** specifies the initial volumetric soil moisture. (units  $\frac{m^3}{m^3}$ )

**Catchment initial soil moisture:** 0.30

**Catchment initial soil temperature:** specifies the initial soil temperature in Kelvin.

**Catchment initial soil temperature:** 290.0

## 8.10 Model output configuration

The output start time is used to define when to begin writing model output. Any value not defined will default to the corresponding LIS start time. The output start time does not affect restart writing. Restart files are written according to the LIS start time and the model restart output interval value.

The output start time is specified in the following format:

Variable	Value	Description
<b>Output start year:</b>	integer 2001 – present	specifying output start year
<b>Output start month:</b>	integer 1 – 12	specifying output start month
<b>Output start day:</b>	integer 1 – 31	specifying output start day
<b>Output start hour:</b>	integer 0 – 23	specifying output start hour
<b>Output start minute:</b>	integer 0 – 59	specifying output start minute
<b>Output start second:</b>	integer 0 – 59	specifying output start second

**Output start year:**  
**Output start month:**  
**Output start day:**  
**Output start hour:**  
**Output start minutes:**  
**Output start seconds:**

This section defines which ALMA variables to write in the model output. Acceptable values are:

Value	Description
0	do not write the variable
1	write the variable

Note that this is a full list of output variables. Not all models support all these variables. You must check the source code to verify that the model you want to run supports the variables that you want to write. Specifying a value of 1 for an unsupported variable does not cause a problem.

```
#Energy balance components
Swnet:      1      # Net Shortwave Radiation (W/m2)
Lwnet:      1      # Net Longwave Radiation (W/m2)
Qle:        1      # Latent Heat Flux (W/m2)
Qh:         1      # Sensible Heat Flux (W/m2)
Qg:         1      # Ground Heat Flux (W/m2)
Qf:         0      # Energy of fusion (W/m2)
Qv:         0      # Energy of sublimation (W/m2)
Qa:         0      # Advective Energy (W/m2)
Qtau:       0      # Momentum flux (N/m2)
DelSurfHeat: 0      # Change in surface heat storage (J/m2)
DelColdCont: 0      # Change in snow cold content (J/m2)

#Water balance components
Snowf:      1      # Snowfall rate (kg/m2s)
Rainf:      1      # Rainfall rate (kg/m2s)
Evap:       1      # Total Evapotranspiration (kg/m2s)
Qs:         1      # Surface runoff (kg/m2s)
Qrec:       0      # Recharge (kg/m2s)
Qsb:        1      # Subsurface runoff (kg/m2s)
Qsm:        0      # Snowmelt (kg/m2s)
Qfz:        0      # Refreezing of water in the snowpack (kg/m2s)
Qst:        0      # Snow throughfall (kg/m2s)
DelSoilMoist: 0     # Change in soil moisture (kg/m2)
DelsWE:     0      # Change in snow water equivalent (kg/m2)
DelSurfStor: 0     # Change in surface water storage (kg/m2)
DelIntercept: 0    # Change in interception storage (kg/m2)

#Surface State Variables
SnowT:      0      # Snow surface temperature (K)
VegT:       0      # Vegetation canopy temperature (K)
BareSoilT:   0      # Temperature of bare soil (K)
AvgSurfT:   1      # Average surface temperature (K)
RadT:       0      # Surface Radiative Temperature (K)
```

```

Albedo:      1      # Surface Albedo (-)
SWE:         1      # Snow Water Equivalent (kg/m2)
SnowDepth:   1      # Snow Depth (m)
SWEVeg:      0      # SWE intercepted by vegetation (kg/m2)
SurfStor:    0      # Surface water storage (kg/m2)

#Subsurface State Variables
SoilMoist:   1      # Average layer soil moisture (kg/m2)
SoilTemp:    1      # Average layer soil temperature (K)
SmLiqFrac:   0      # Average layer fraction of liquid moisture (-)
SmFrozFrac:  0      # Average layer fraction of frozen moisture (-)
SoilWet:     0      # Total soil wetness (-)

#Evaporation components
PotEvap:     0      # Potential Evapotranspiration (kg/m2s)
ECanop:       0      # Interception evaporation (kg/m2s)
TVeg:        0      # Vegetation transpiration (kg/m2s)
ESoil:        0      # Bare soil evaporation (kg/m2s)
EWater:       0      # Open water evaporation (kg/m2s)
RootMoist:   0      # Root zone soil moisture (kg/m2)
CanopInt:    1      # Total canopy water storage (kg/m2)
EvapSnow:    0      # Snow evaporation (kg/m2s)
SubSnow:     0      # Snow sublimation (kg/m2s)
SubSurf:    0      # Sublimation of the snow free area (kg/m2s)
ACond:       0      # Aerodynamic conductance

#Parameters
LandMask:    1      # Land Mask (0 == Water, 1 == Land)

```

## 9 Test Cases

This section describes how to use the test cases included in the source code.

The layout of the *testcases* directory matches the layout of the top-level *src* directory. For example, LIS contains support for processing GDAS forcing data. These routines are in *src/baseforcing/gdas*. The test case for GDAS is in *src/testcases/baseforcing/gdas*.

These test case sub-directories contain several files to help you test LIS. For example, the *src/testcases/baseforcing/gdas* test case contains: *README*, *lis.config*, *output.ctl*

The file, *README*, contains instructions on how to run the test case. The file, *lis.config*, is a configuration file to set the test case. The file, *output.ctl*, is a GrADS descriptor file. This file is used with GrADS to plot the output data. You may also read this file to obtain metadata regarding the structure of the output files. This metadata is useful in helping you plot the output using a different program.

## 10 Output Data Processing

This section describes how to process the generated output.

The output data-sets created by running the LIS executable are written into sub-directories of the `$WORKING/run/OUTPUT/` directory (created at run-time). These sub-directories are named such as `EXP999`.

The output data consists of ASCII text files and model output in binary format.

For example, assume that you performed a “1/4 deg Noah with GEOS forcing” using a single nest, simulation for 11 June 2001, with an experiment code value of 999, and writing Fortran binary output as a 2-D array.

This run will produce a `$WORKING/run/OUTPUT/EXP999/` directory. This directory will contain:

File Name	Synopsis
Noahstats.dat	Statistical summary of output
NOAH	Directory containing output data

The `NOAH` directory will contain sub-directories of the form `YYYY/YYYYMMDD`, where `YYYY` is a 4-digit year and `YYYYMMDD` is a date written as a 4-digit year, 2-digit month and a 2-digit day; both corresponding to the running dates of the simulation.

For this example, `NOAH` will contain a `2001/20010611` sub-directory.

Its contents are the output files generated by the executable. They are:

```
200106110000.d01.gs4r  
200106110300.d01.gs4r  
200106110600.d01.gs4r  
200106110900.d01.gs4r  
200106111200.d01.gs4r  
200106111500.d01.gs4r  
200106111800.d01.gs4r  
200106112100.d01.gs4r
```

Note, each file-name contains a date-stamp marking the year, month, day, hour, and minute that the data corresponds to. The output data files for other land surface models are similar.

The actual contents of the output files depend on the settings in the `lis.config` configuration file.

The generated output can be written in a Fortran binary, GRIB, or NetCDF format. See Section ?? for more details.

The LIS team uses GrADS to visualize the output data. GrADS has the capability to handle binary, GRIB, and NetCDF data. See the documentation for

GrADS at (<http://www.iges.org/grads/grads.html>) for more information. The `$WORKING/src/utils/grads/` directory contains a utility for creating a GrADS descriptor file. See the `$WORKING/src/utils/grads/README` for more information. Note that GRIB and NetCDF data files may be inspected by using the utilities `wgrib` and `ncdump`, respectively.

The generated output can be written in a 2-D grid format or as a 1-d vector. See Section ?? for more details. If written as a 1-d vector, the output must be converted into a 2-d grid before it can be visualized.

## 11 Retrieving Sample Output Data

This section describes how to obtain sample output data-sets for testing the LIS executable.

The LIS team has created many test cases for testing various features and functionality of LIS.

Please read Section 9 for more details about the test cases.

### 11.1 Downloading the Output Data-sets

To obtain sample output data-sets for LIS revision 5.0:

1. Go to LIS’ “Public Release Home Page”  
Go to <http://lis.gsfc.nasa.gov/>  
Follow the “Source Codes” link.  
Follow the “LIS 5.0 Code Release” link.
2. From LIS’ “Public Release Home Page”  
Follow the instructions in the “Output Data” section.

### 11.2 Example

For example, sample output data for the “Noah LSM TEST” contains:

```
OUTPUT/EXP111/NOAHstats.d01.stats  
OUTPUT/EXP111/NOAH/2002/20021029/200210290300.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021029/200210290600.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021029/200210290900.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021029/200210291200.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021029/200210291500.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021029/200210291800.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021029/200210292100.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021030/200210300000.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021030/LIS.E111.200210300000.d01.Noahrst  
OUTPUT/EXP111/NOAH/2002/20021030/200210300300.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021030/200210300600.d01.gs4r  
OUTPUT/EXP111/NOAH/2002/20021030/200210300900.d01.gs4r
```

```
OUTPUT/EXP111/NOAH/2002/20021030/200210301200.d01.gs4r
OUTPUT/EXP111/NOAH/2002/20021030/200210301500.d01.gs4r
OUTPUT/EXP111/NOAH/2002/20021030/200210301800.d01.gs4r
OUTPUT/EXP111/NOAH/2002/20021030/200210302100.d01.gs4r
OUTPUT/EXP111/NOAH/2002/20021031/200210310000.d01.gs4r
OUTPUT/EXP111/NOAH/2002/20021031/LIS.E111.200210310000.d01.Noahrst
OUTPUT/EXP111/NOAH/2002/20021031/LIS.E111.200210310100.d01.Noahrst
```

The file, *OUTPUT/EXP111/NOAHstats.d01.stats*, contains statistics from the run.

The files labelled like *OUTPUT/EXP111/NOAH/2002/20021029/200210290300.d01.gs4r* contain the output from the run. Read the *output.ctl* file contained in the appropriate *testcases* sub-directory of the LIS source code for metadata pertaining to these output files.

The files labelled like

*OUTPUT/EXP111/NOAH/2002/20021031/LIS.E111.200210310000.d01.Noahrst* are restart files. They may be used to continue or restart a run. The data are valid for the date and time indicated by the date-stamp in the file name. For example, the restart data in this file, *OUTPUT/EXP111/NOAH/2002/20021031/LIS.E111.200210310000.d01.Noahrst*, are valid for 2002-10-31T00:00:00.

These output data files are large and require post-processing before reading them, see Section 10.

## A Cylindrical Lat/Lon Domain Example

This section describes how to compute the values for the run domain and param domain sections on a cylindrical lat/lon projection.

First, we shall generate the values for the parameter data domain. LIS' parameter data is defined on a Latitude/Longitude grid, from  $-180$  to  $180$  degrees longitude and from  $-60$  to  $90$  degrees latitude.

For this example, consider running at  $1/4$  deg resolution. The coordinates of the south-west and the north-east points are specified at the grid-cells' centers. Here the south-west grid-cell is given by the box  $(-180, -60), (-179.750, -59.750)$ . The center of this box is  $(-179.875, -59.875)$ .<sup>1</sup>

```
param domain lower left lat: -59.875
param domain lower left lon: -179.875
```

The north-east grid-cell is given by the box  $(179.750, 89.750), (180, 90)$ . Its center is  $(179.875, 89.875)$ .

```
param domain upper right lat: 89.875
param domain upper right lon: 179.875
```

Setting the resolution ( $0.25$  deg) gives

```
param domain resolution dx: 0.25
param domain resolution dy: 0.25
```

And this completely defines the parameter data domain.

Next, we shall generate the values for the running domain.

If you wish to run over the whole domain defined by the parameter data domain then you simply set the values defined in the parameter domain section in the run domain section. This gives:

```
run domain lower left lat: -59.875
run domain lower left lon: -179.875
run domain upper right lat: 89.875
run domain upper right lon: 179.875
run domain resolution dx: 0.25
run domain resolution dy: 0.25
```

Now say you wish to run only over the region given by  $(-97.6, 27.9), (-92.9, 31.9)$ . Since the running domain is a sub-set of the parameter domain, it is also a Latitude/Longitude domain at  $1/4$  deg. resolution. Thus,

```
run domain resolution dx: 0.25
run domain resolution dy: 0.25
```

Now, since the running domain must fit onto the parameter domain, the desired running region must be expanded from  $(-97.6, 27.9), (-92.9, 31.9)$  to  $(-97.75, 27.75), (-92.75, 32.0)$ . The south-west grid-cell for the running domain is the box  $(-97.75, 27.75), (-97.5, 28.0)$ . Its center is  $(-97.625, 27.875)$ ; giving

---

<sup>1</sup>Note, these coordinates are ordered (longitude, latitude).

```
run domain lower left lat: 27.875  
run domain lower left lon: -97.625
```

The north-east grid-cell for the running domain is the box  $(-93, 31.75), (-92.75, 32.0)$ . Its center is  $(-92.875, 31.875)$ ; giving

```
run domain upper right lat: 31.875  
run domain upper right lon: -92.875
```

This completely defines the running domain.

Note, the LIS project has defined 5 km resolution to be 0.05 deg. and 1 km resolution to be 0.01 deg. If you wish to run at 5 km or 1 km resolution, redo the above example to compute the appropriate grid-cell values.

See Figure 1 for an illustration of adjusting the running grid. See Figures 2 and 3 for an illustration of the south-west and north-east grid-cells.

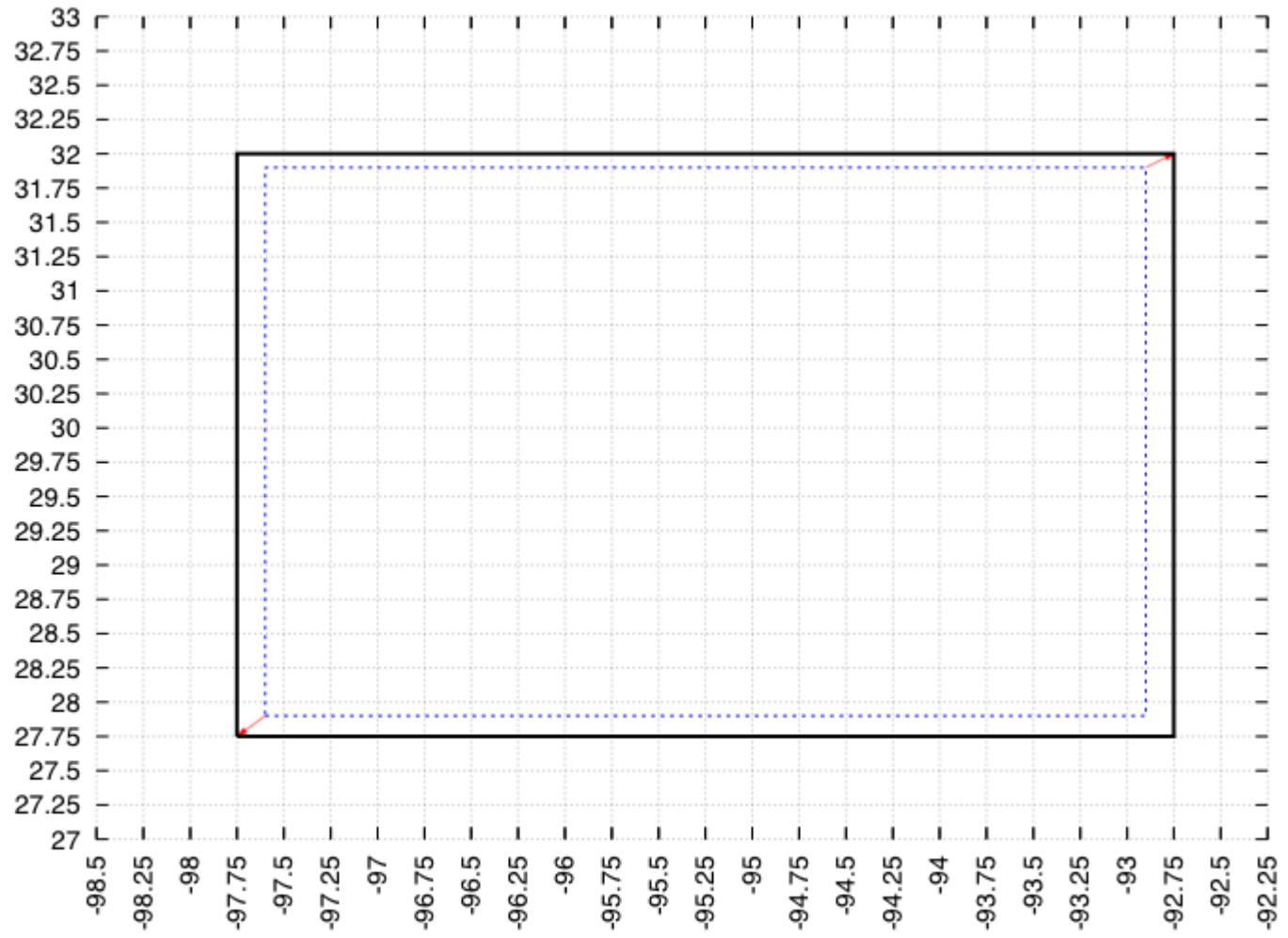


Figure 1: Illustration showing how to fit the desired running grid onto the actual grid

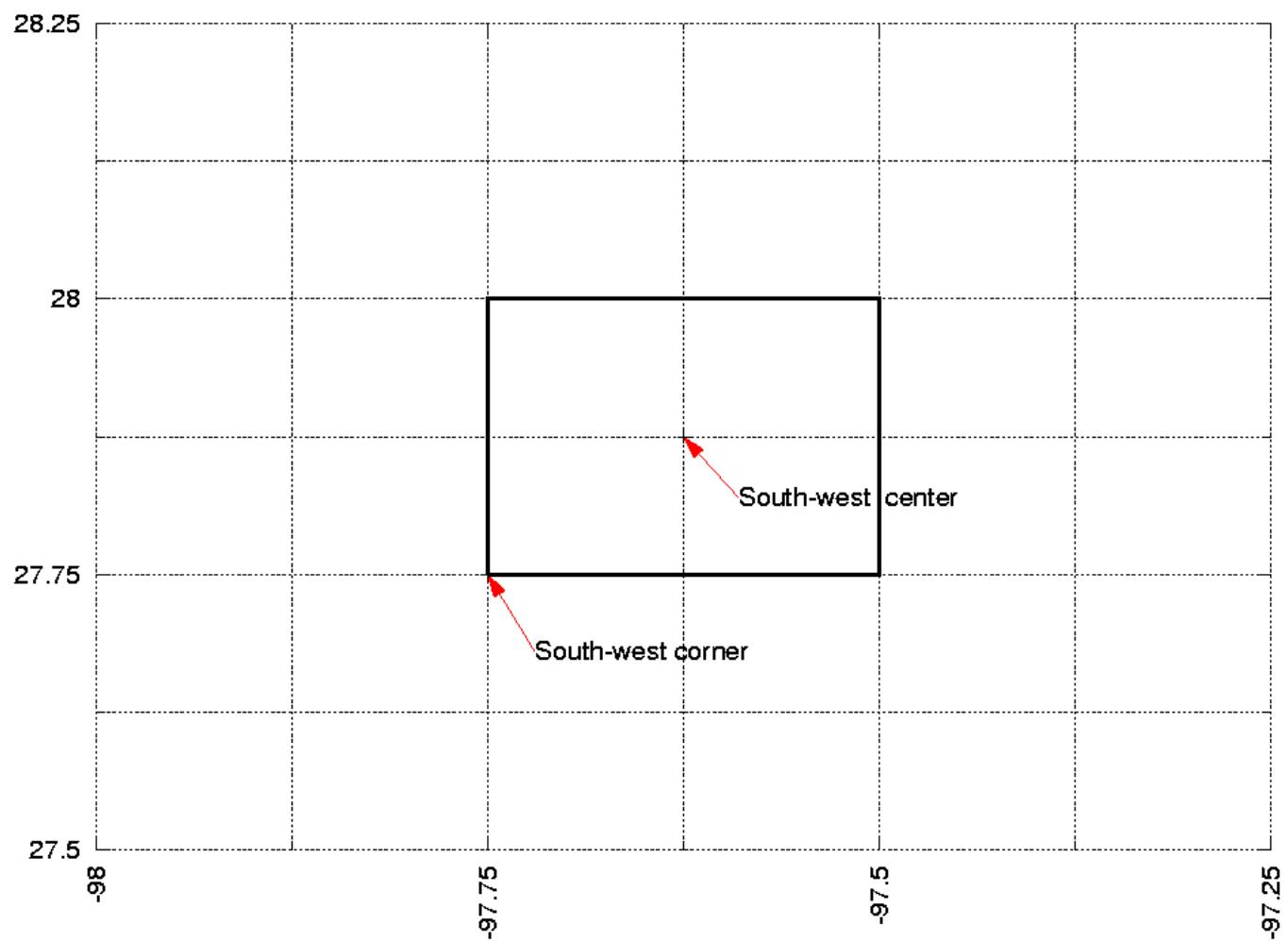


Figure 2: Illustration showing the south-west grid-cell corresponding to the example in Section A

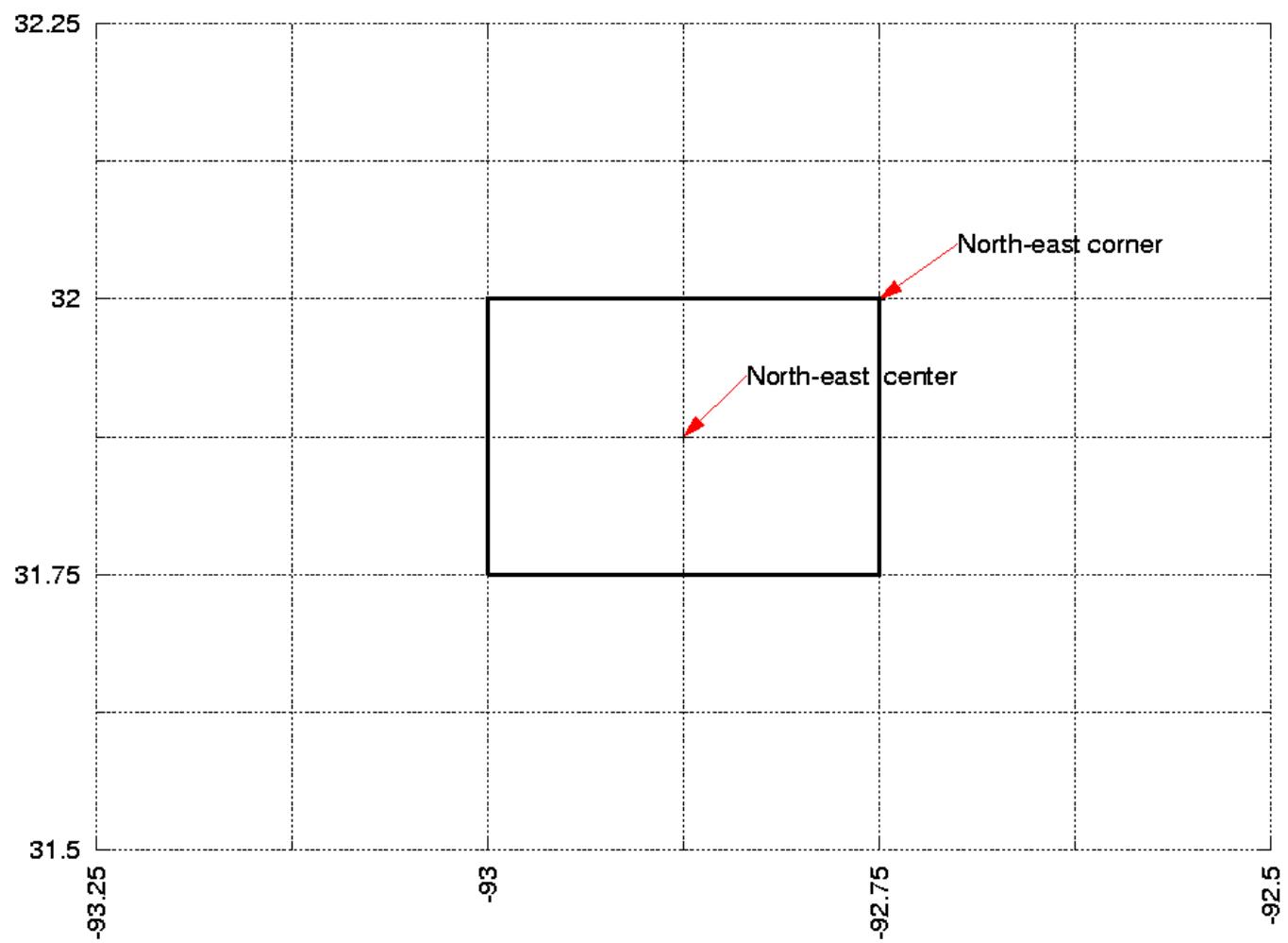


Figure 3: Illustration showing the north-east grid-cell corresponding to the example in Section A

## B Polar Stereographic Domain Example

This section describes how to compute the values for the run domain and param domain sections on a polar stereographic projection.

STUB!

## C Gaussian Domain Example

This section describes how to compute the values for the run domain and param domain sections on a Gaussian projection.

First, we shall generate the values for the parameter data domain. LIS' Gaussian parameter data is defined from  $-180$  to  $180$  degrees longitude and from  $-90$  to  $90$  degrees latitude. Note that the first longitude point is at  $0$ .

The parameter domain must be specified as follows:

```
param domain first grid point lat:      -89.27665
param domain first grid point lon:      0.0
param domain last grid point lat:       89.27665
param domain last grid point lon:      -0.9375
param domain resolution dlon:          0.9375
param domain number of lat circles:    95
```

Next, we shall generate the values for the running domain.

If you wish to run over the whole domain defined by the parameter data domain then you simply set the values defined in the parameter domain section in the run domain section. This gives:

```
run domain first grid point lat:      -89.27665
run domain first grid point lon:      0.0
run domain last grid point lat:       89.27665
run domain last grid point lon:      -0.9375
run domain resolution dlon:          0.9375
run domain number of lat circles:    95
```

If you wish to run over a sub-domain, then you must choose longitude and latitude values that correspond to the T126 Gaussian projection. Tables of acceptable longitude and latitude values are found below.

Now say you wish to run only over the region given by  $(-97.6, 27.9), (-92.9, 31.9)$ . Since the running domain must fit on the T126 Gaussian grid, the running domain must be expanded to  $(-98.4375, 27.87391), (-91.875, 32.59830)$ . Thus the running domain specification is:

```
run domain first grid point lat:      27.87391
run domain first grid point lon:      -98.4375
run domain last grid point lat:       32.59830
run domain last grid point lon:      -91.875
run domain resolution dlon:          0.9375
run domain number of lat circles:    95
```

Table 1: Acceptable longitude values

0.000000	0.937500	1.875000	2.812500	3.750000
4.687500	5.625000	6.562500	7.500000	8.437500
9.375000	10.312500	11.250000	12.187500	13.125000
14.062500	15.000000	15.937500	16.875000	17.812500
18.750000	19.687500	20.625000	21.562500	22.500000
23.437500	24.375000	25.312500	26.250000	27.187500
28.125000	29.062500	30.000000	30.937500	31.875000
32.812500	33.750000	34.687500	35.625000	36.562500
37.500000	38.437500	39.375000	40.312500	41.250000
42.187500	43.125000	44.062500	45.000000	45.937500
46.875000	47.812500	48.750000	49.687500	50.625000
51.562500	52.500000	53.437500	54.375000	55.312500
56.250000	57.187500	58.125000	59.062500	60.000000
60.937500	61.875000	62.812500	63.750000	64.687500
65.625000	66.562500	67.500000	68.437500	69.375000
70.312500	71.250000	72.187500	73.125000	74.062500
75.000000	75.937500	76.875000	77.812500	78.750000
79.687500	80.625000	81.562500	82.500000	83.437500
84.375000	85.312500	86.250000	87.187500	88.125000
89.062500	90.000000	90.937500	91.875000	92.812500
93.750000	94.687500	95.625000	96.562500	97.500000
98.437500	99.375000	100.312500	101.250000	102.187500
103.125000	104.062500	105.000000	105.937500	106.875000
107.812500	108.750000	109.687500	110.625000	111.562500
112.500000	113.437500	114.375000	115.312500	116.250000
117.187500	118.125000	119.062500	120.000000	120.937500
121.875000	122.812500	123.750000	124.687500	125.625000
126.562500	127.500000	128.437500	129.375000	130.312500
131.250000	132.187500	133.125000	134.062500	135.000000
135.937500	136.875000	137.812500	138.750000	139.687500
140.625000	141.562500	142.500000	143.437500	144.375000
145.312500	146.250000	147.187500	148.125000	149.062500
150.000000	150.937500	151.875000	152.812500	153.750000
154.687500	155.625000	156.562500	157.500000	158.437500
159.375000	160.312500	161.250000	162.187500	163.125000
164.062500	165.000000	165.937500	166.875000	167.812500
168.750000	169.687500	170.625000	171.562500	172.500000
173.437500	174.375000	175.312500	176.250000	177.187500
178.125000	179.062500	180.000000	-179.062500	-178.125000

-177.187500	-176.250000	-175.312500	-174.375000	-173.437500
-172.500000	-171.562500	-170.625000	-169.687500	-168.750000
-167.812500	-166.875000	-165.937500	-165.000000	-164.062500
-163.125000	-162.187500	-161.250000	-160.312500	-159.375000
-158.437500	-157.500000	-156.562500	-155.625000	-154.687500
-153.750000	-152.812500	-151.875000	-150.937500	-150.000000
-149.062500	-148.125000	-147.187500	-146.250000	-145.312500
-144.375000	-143.437500	-142.500000	-141.562500	-140.625000
-139.687500	-138.750000	-137.812500	-136.875000	-135.937500
-135.000000	-134.062500	-133.125000	-132.187500	-131.250000
-130.312500	-129.375000	-128.437500	-127.500000	-126.562500
-125.625000	-124.687500	-123.750000	-122.812500	-121.875000
-120.937500	-120.000000	-119.062500	-118.125000	-117.187500
-116.250000	-115.312500	-114.375000	-113.437500	-112.500000
-111.562500	-110.625000	-109.687500	-108.750000	-107.812500
-106.875000	-105.937500	-105.000000	-104.062500	-103.125000
-102.187500	-101.250000	-100.312500	-99.375000	-98.437500
-97.500000	-96.562500	-95.625000	-94.687500	-93.750000
-92.812500	-91.875000	-90.937500	-90.000000	-89.062500
-88.125000	-87.187500	-86.250000	-85.312500	-84.375000
-83.437500	-82.500000	-81.562500	-80.625000	-79.687500
-78.750000	-77.812500	-76.875000	-75.937500	-75.000000
-74.062500	-73.125000	-72.187500	-71.250000	-70.312500
-69.375000	-68.437500	-67.500000	-66.562500	-65.625000
-64.687500	-63.750000	-62.812500	-61.875000	-60.937500
-60.000000	-59.062500	-58.125000	-57.187500	-56.250000
-55.312500	-54.375000	-53.437500	-52.500000	-51.562500
-50.625000	-49.687500	-48.750000	-47.812500	-46.875000
-45.937500	-45.000000	-44.062500	-43.125000	-42.187500
-41.250000	-40.312500	-39.375000	-38.437500	-37.500000
-36.562500	-35.625000	-34.687500	-33.750000	-32.812500
-31.875000	-30.937500	-30.000000	-29.062500	-28.125000
-27.187500	-26.250000	-25.312500	-24.375000	-23.437500
-22.500000	-21.562500	-20.625000	-19.687500	-18.750000
-17.812500	-16.875000	-15.937500	-15.000000	-14.062500
-13.125000	-12.187500	-11.250000	-10.312500	-9.375000
-8.437500	-7.500000	-6.562500	-5.625000	-4.687500
-3.750000	-2.812500	-1.875000	-0.937500	

Table 2: Acceptable latitude values

-89.27665	-88.33975	-87.39729	-86.45353	-85.50930
-84.56487	-83.62028	-82.67562	-81.73093	-80.78618
-79.84142	-78.89662	-77.95183	-77.00701	-76.06219
-75.11736	-74.17252	-73.22769	-72.28285	-71.33799
-70.39314	-69.44830	-68.50343	-67.55857	-66.61371
-65.66885	-64.72399	-63.77912	-62.83426	-61.88939
-60.94452	-59.99965	-59.05478	-58.10991	-57.16505
-56.22018	-55.27531	-54.33043	-53.38556	-52.44069
-51.49581	-50.55094	-49.60606	-48.66119	-47.71632
-46.77144	-45.82657	-44.88169	-43.93681	-42.99194
-42.04707	-41.10219	-40.15731	-39.21244	-38.26756
-37.32268	-36.37781	-35.43293	-34.48805	-33.54317
-32.59830	-31.65342	-30.70854	-29.76366	-28.81879
-27.87391	-26.92903	-25.98415	-25.03928	-24.09440
-23.14952	-22.20464	-21.25977	-20.31489	-19.37001
-18.42513	-17.48025	-16.53537	-15.59050	-14.64562
-13.70074	-12.75586	-11.81098	-10.86610	-9.921225
-8.976346	-8.031467	-7.086589	-6.141711	-5.196832
-4.251954	-3.307075	-2.362196	-1.417318	-0.4724393
0.4724393	1.417318	2.362196	3.307075	4.251954
5.196832	6.141711	7.086589	8.031467	8.976346
9.921225	10.86610	11.81098	12.75586	13.70074
14.64562	15.59050	16.53537	17.48025	18.42513
19.37001	20.31489	21.25977	22.20464	23.14952
24.09440	25.03928	25.98415	26.92903	27.87391
28.81879	29.76366	30.70854	31.65342	32.59830
33.54317	34.48805	35.43293	36.37781	37.32268
38.26756	39.21244	40.15731	41.10219	42.04707
42.99194	43.93681	44.88169	45.82657	46.77144
47.71632	48.66119	49.60606	50.55094	51.49581
52.44069	53.38556	54.33043	55.27531	56.22018
57.16505	58.10991	59.05478	59.99965	60.94452
61.88939	62.83426	63.77912	64.72399	65.66885
66.61371	67.55857	68.50343	69.44830	70.39314
71.33799	72.28285	73.22769	74.17252	75.11736
76.06219	77.00701	77.95183	78.89662	79.84142
80.78618	81.73093	82.67562	83.62028	84.56487
85.50930	86.45353	87.39729	88.33975	89.27665

## D Lambert Conformal Domain Example

This section describes how to compute the values for the run domain and param domain sections on a Lambert conformal projection.

STUB!

## E Mercator Domain Example

This section describes how to compute the values for the run domain and param domain sections on a Mercator projection.

STUB!

## F Polar Global Domain Example

This section describes how to compute the values for the run domain and param domain sections on a polar global projection.

This projection is not supported!

STUB!

## G Makefile

```
# Set up special characters
include ./configure.lis

null  :=
space := $(null) $(null)
doctool :=../utils/docsgen.sh

# Check for directory in which to put executable
ifeq ($(MODEL_EXEDIR),$(null))
MODEL_EXEDIR := .
endif

# Check for name of executable
ifeq ($(EXENAME),$(null))
EXENAME := LIS
endif

# Check if COUPLED is defined in "misc.h"
# Ensure that it is defined and not just "undef COUPLED" set in file
CPLD := $(shell grep COUPLED misc.h)
ifeq ($(findstring define,$(CPLD)), define)
    CPLD := TRUE
else
    CPLD := FALSE
endif

# Determine platform
UNAMES := $(shell uname -s)
UMACHINE := $(shell uname -m)

# Load dependency search path.
dirs := . $(shell cat Filepath)
ifeq ($(CPLD), FALSE)
dirs := $(dirs) ../offline
endif
# Set cpp search path, include netcdf
cpp_dirs := $(dirs) $(INC_MPI)
cpp_path := $(foreach dir,$(cpp_dirs),-I$(dir)) # format for command line

# Expand any tildes in directory names. Change spaces to colons.
VPATH    := $(foreach dir,$(cpp_dirs),$(wildcard $(dir)))
VPATH    := $(subst $(space),:,$(VPATH))

#-----
```

```

# Primary target: build the model
#-----

# Get list of files and determine objects and dependency files
FIND_FILES = $(wildcard $(dir)/*.F $(dir)/*.f $(dir)/*.F90 $(dir)/*.c $(dir)/*.f)
FILES      = $(foreach dir, $(dirs), $(FIND_FILES))
SOURCES   := $(sort $(notdir $(FILES)))
DEPS      := $(addsuffix .d, $(basename $(SOURCES)))
OBJS      := $(addsuffix .o, $(basename $(SOURCES)))
DOCS      := $(addsuffix .tex, $(basename $(SOURCES)))

$(MODEL_EXEDIR)/$(EXENAME): $(OBJS)
    $(FC) -o $@ $(OBJS) $(FOPTS) $(LDFLAGS)

LIBTARGET   = explis
TARGETDIR   = ./$(LIBTARGET):
gmake -i -r lis_contrib ; \
$(AR) ../../main/libwrflib.a $(OBJS) ; \

lis_contrib : $(OBJS)

debug: $(OBJS)
echo "FFLAGS: $(FFLAGS)"
echo "LDFLAGS: $(LDFLAGS)"
echo "OBJS: $(OBJS)"

*****
***** Architecture-specific flags and rules*****
*****

FFLAGS      += $(cpp_path)

.SUFFIXES:
.SUFFIXES: .f .f90 .F90 .c .o

.f.o:
$(FC77) $(FFLAGS77) $<
.F90.o:
$(FC) $(FFLAGS) $<
.c.o:
$(CC) -c $(cpp_path) $(CFLAGS) $<
.f90.o:
$(FC) $(FFLAGS) $<

```

```

RM := rm
# Add user defined compiler flags if set, and replace FC if USER option set.
FFLAGS += $(USER_FFLAGS)
ifeq ($(USER_FC),$(null))
FC := $(USER_FC)
endif

clean:
$(RM) -f *.o *.mod *.stb $(MODEL_EXEDIR)/$(EXENAME)
realclean:
$(RM) -f *.o *.d *.mod *.stb $(MODEL_EXEDIR)/$(EXENAME)
doc:
$(doctool)

#-----
#!!!!!!!!!!!!!!DO NOT EDIT BELOW THIS LINE.!!!!!!!!!!!!!!
#-----

# These rules cause a dependency file to be generated for each source
# file. It is assumed that the tool "makdep" (provided with this
# distribution in clm2/tools/makdep) has been built and is available in
# the user's $PATH. Files contained in the clm2 distribution are the
# only files which are considered in generating each dependency. The
# following filters are applied to exclude any files which are not in
# the distribution (e.g. system header files like stdio.h).
#
# 1) Remove full paths from dependencies. This means gnumake will not break
#    if new versions of files are created in the directory hierarchy
#    specified by VPATH.
#
# 2) Because of 1) above, remove any file dependencies for files not in the
#    clm2 source distribution.
#
# Finally, add the dependency file as a target of the dependency rules. This
# is done so that the dependency file will automatically be regenerated
# when necessary.
#
#     i.e. change rule
#       make.o : make.c make.h
#       to:
#       make.o make.d : make.c make.h
#-----
DEPGEN := ./MAKDEP/makdep -s F
%.d : %.c
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
%.d : %.f

```

```

@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
%.d : %.F90
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
%.d : %.F
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
#
# if goal is clean or realclean then don't include .d files
# without this is a hack, missing dependency files will be created
# and then deleted as part of the cleaning process
#
INCLUDE_DEPS=TRUE
ifeq ($(MAKECMDGOALS), realclean)
    INCLUDE_DEPS=false
endif
ifeq ($(MAKECMDGOALS), clean)
    INCLUDE_DEPS=false
endif

ifeq ($(INCLUDE_DEPS), TRUE)
-include $(DEPS)
endif

```

## H configure.lis

This is a sample *configure.lis* file used for compiling LIS on an IBM AIX system.

```
FC          = mpxlf90_r
FC77        = mpxlf_r
LD          = mpxlf90_r
CC          = mpcc_r
INC_NETCDF  = /usrx/local/netcdf/include
LIB_NETCDF  = /usrx/local/netcdf
LIB_MPI     = /usr/lpp/ppe.poe/lib
INC_MPI     = /usr/lpp/ppe.poe/include
LIB_ESMF    = /u/home/sujay/esmf_2_2_0rp1/lib/lib0/AIX.default.64.default
MOD_ESMF    = /u/home/sujay/esmf_2_2_0rp1/mod/mod0/AIX.default.64.default
CFLAGS      = -c -w -g -q64 -qcpluscmt -DSPMD
FFLAGS      = -I$(INC_MPI) -I$(MOD_ESMF) -c -g -qkeepparm \
              -qsuffix=f=F90:cpp=F90 -q64 -WF,-DAIX,-DSPMD
FFLAGS77    = -I$(INC_MPI) -I$(MOD_ESMF) -c -g -qkeepparm \
              -qsuffix=f=f:cpp=F90 -q64 -WF,-DAIX,-DSPMD
LDFLAGS     = -q64 -bmap:map -bloadmap:lm -lmass \
              ../lib/w3lib/libw3.a \
              ../lib/read_grib.aix/readgrib.a \
              ../lib/grib/griblib.a \
              -L$(LIB_ESMF) -lesmf -lnetcdf_stubs -lC_r
```

# I READ GRIB - Information and Instructions

Thanks to Kristi Arsenault for putting this together

## Caveat:

This is a package of subroutines to read GRIB-formatted data. It is still under continuous development. It won't be able to read every GRIB dataset you give it, but it will read a good many.

- Kevin W. Manning  
NCAR/MMM  
Summer 1998, and continuing

The main user interfaces are:

**SUBROUTINE GRIBGET(NUNIT, IERR)** - Read a single GRIB record from UNIX file-descriptor NUNIT into array GREC. No unpacking of any header or data values is performed.

**SUBROUTINE GRIBREAD(NUNIT, DATA, NDATA, IERR)** - Read a single GRIB record from UNIX file-descriptor NUNIT, and unpack all header and data values into the appropriate arrays.

**SUBROUTINE GRIBHEADER(IERR)** - Unpack the header of a GRIB record

**SUBROUTINE GRIBDATA(DATARRAY, NDAT)** - Unpack the data in a GRIB record into array DATARRAY

**SUBROUTINE GRIBPRINT(ISEC)** - Print the header information from GRIB section ISEC.

**SUBROUTINE GET\_SEC1(KSEC1)** - Return the header information from Section 1.

**SUBROUTINE GET\_SEC2(KSEC2)** - Return the header information from Section 2.

**SUBROUTINE GET\_GRIDINFO(IGINFO, GINFO)** - Return the grid information of the previously-unpacked GRIB header.

**C-ROUTINE – COPEN(UNIT, NUNIT, NAME, MODE, ERR, OFLAG)** - Opens the GRIB file for reading later by the other routines.

## I.1 SUBROUTINE GRIBGET (NUNIT, IERR)

- Read a single GRIB record from UNIX file-descriptor NUNIT into array GREC. No unpacking of any header or data values is performed.

- NOTE!: Intrinsic parameter, ied, identifies type of GRIB edition of GRIB file trying to open. Below are the codes (also see Table3):

- If IED == 1, then GRIB Edition 1 has the size of the whole GRIB record right up front.
- If IED == 0, then GRIB Edition 0 does not include the total size, so we have to sum up the sizes of the individual sections

- If IED > 1, then STOP, if not GRIB Edition 0 or 1

Input: NUNIT (integer): C unit number to read from. This should already be open.

Output: IERR (integer): Error flag, Non-zero means there was a problem with the read.

Side Effects: The array GREC is allocated, and filled with one GRIB record. The C unit pointer is moved to the end of the GRIB record just read.

## **I.2 SUBROUTINE GRIBREAD (NUNIT, DATA, NDATA, IERR)**

- Read a single GRIB record from UNIX file-descriptor, NUNIT, unpack all header and data values into the appropriate arrays, and fill the allocatable array, DATARRAY(:).

Input: NUNIT (integer): C Unit to read from.

NDATA (integer): Size of array DATA (Should be  $\leq$  NDAT as computed herein.)

Output: DATA (real): The unpacked data array (dimension size of NDATA)  
IERR(integer): Error flag, non-zero means there was a problem.

Side Effects: Header arrays SEC0, SEC1, SEC2, SEC3, SEC4, XEC4, INFOGRID and INFOGRID are filled.

The BITMAP array is filled.

The C unit pointer is advanced to the end of the GRIB record.

## **I.3 SUBROUTINE GRIBHEADER (IERR)**

Unpack the header of a GRIB record

IERR non-zero means there was a problem unpacking the grib header

IERR (integer)

## **I.4 SUBROUTINE GRIBDATA (DATARRAY, NDAT)**

- Read and unpack the data in a GRIB record into array, DATARRAY

Input: NDAT (integer): The size of the data array we expect to unpack.

Output: DATARRAY (real): The unpacked data from the GRIB record (dimension size of NDAT)

Side Effects: - STOP – if it cannot unpack the data.

## I.5 SUBROUTINE GRIBPRINT (ISEC)

Print the header information from GRIB section ISEC.

ISEC Information can be found in the section I.10

## I.6 SUBROUTINE GET\_SEC1 (KSEC1)

- Return the header information from GRIB Section 1 (see I.10, TABLE 4).
- Return the GRIB Section 1 header information, which has already been unpacked by subroutine GRIBHEADER.
- KSEC1 (integer :: dimension(100))

## I.7 SUBROUTINE GET\_SEC2 (KSEC2)

- Return the header information from GRIB Section 2 (see I.10, TABLE 5).
- Return the GRIB Section 2 header information, which has already been unpacked by subroutine GRIBHEADER.
- KSEC2 (integer :: dimension(10))

## I.8 SUBROUTINE GET\_GRIDINFO (IGINFO, GINFO)

- Return the grid information of the previously-unpacked GRIB header.
  - IGINFO (integer :: dimension(40))
  - GINFO (real :: dimension(40))
- \*\* NOTE IGINFO and GINFO contain equivalent information, except that IGINFO is the integer form and GINFO is the real form.

## I.9 C-ROUTINE COPEN (UNIT, NUNIT, NAME, MODE, ERR, OFLAG)

- Opens the GRIB file for reading later by the other subroutines

UNIT = Fortran unit number (integer)

NUNIT = UNIX file descriptor associated with 'unit' (integer)

NAME = UNIX file name (character (len=120) )

MODE = 0 : write only - file will be created if it doesn't exist, - otherwise will be rewritten (integer)  
= 1 : read only

= 2 : read/write

ERR = 0 : no error opening file (integer)  
 $\neq 0$  : Error opening file

OFLAG = 0 : file name printed (no errors printed) (integer)  
 $> 0$  : file name printed and errors are printed  
 $< 0$  : no print at all (not even errors)

## I.10 SEC Header Array Information Tables

Please refer to <http://www.wmo.ch/web/www/WDM/Guides/Guide-binary-2.html> for additional GRIB1 header information

Table 5: SEC2: GRIB Header Section 2 information

Octet	GDS Content	
1-3	Length of GRIB Section 2 (in octets)	
4	Number of vertical-coordinate parameters	
5	Starting-point of the list of vertical-coordinate parameters	
6	Data-representation type (i.e., grid type) See GRIB Table 6 0 = Latitude/Longitude grid 3 = Lambert-conformal grid. 5 = Polar-stereographic grid.	
if(sec2(4)==0) then Lat/lon grid		
INFOGRID	Octet	GDS Content
1	7-8	I Dimension of the grid
2	9-10	J Dimension of the grid
3	11-13	Starting Latitude of the grid.
4	14-16	Starting Longitude of the grid.
5	17	Resolution and component flags.
6	18-20	Ending latitude of the grid.
7	21-23	Ending longitude of the grid.
8	24-25	Longitudinal increment.
9	26-27	Latitudinal increment.
10	28	Scanning mode (bit 3 from Table 8)
21	28	Iscan sign (+1/-1) (bit 1 from Table 8)
22	28	Jscan sign (+1/-1) (bit 2 from Table 8)
if(sec2(4)==1) then mercator grid		
INFOGRID	Octet	GDS Content
1	7-8	I Dimension of the grid

2	9-10	J Dimension of the grid
3	11-13	Starting Latitude of the grid.
4	14-16	Starting Longitude of the grid.
5	17	Resolution and component flags.
6	18-20	Ending latitude of the grid.
7	21-23	Ending longitude of the grid.
8	24-26	LATIN- The latitude(s) at which Mercator projection cylinder intersects the earth
9	27	Reserved (set to 0)
10	28	Scanning mode (bit 3 from Table 8)
11		True Lat
21	29-31	Iscan sign (+1/-1) (bit 1 from Table 8)
22	32-34	Jscan sign (+1/-1) (bit 2 from Table 8)
if(sec2(4)==3) then Lambert Conformal grid		
INFOGRID	Octet	GDS Content
1	7-8	I Dimension of the grid
2	9-10	J Dimension of the grid
3	11-13	Starting Latitude of the grid.
4	14-16	Starting Longitude of the grid.
5	17	Resolution and component flags.
6	18-20	Center longitude of the projection.
7	21-23	Grid-spacing in the I direction
8	24-26	Grid-spacing in the J direction
9	27	Projection center
10	28	Scanning mode (bit 3 from Table 8)
11	29-31	First TRUELAT value.
12	32-34	Second TRUELAT value.
13	35-37	Latitude of the southern pole
14	38-40	Longitude of the southern pole
21	41	Iscan sign (+1/-1) (bit 1 from Table 8)
22	42	Jscan sign (+1/-1) (bit 2 from Table 8)
if(sec2(4)==4) then Gaussian grid		
INFOGRID	Octet	GDS Content
1	7-8	I Dimension of the grid
2	9-10	J Dimension of the grid
3	11-13	Starting Latitude of the grid.
4	14-16	Starting Longitude of the grid.
5	17	Resolution and component flags.
6	18-20	Ending latitude of the grid.
7	21-23	Ending longitude of the grid.
8	24-25	Longitudinal increment.
9	26-27	Latitudinal increment.
10	28	Scanning mode (bit 3 from Table 8)
21	28	Iscan sign (+1/-1) (bit 1 from Table 8)

22	28	Jscan sign (+1/-1) (bit 2 from Table 8)
INFOGRID	if(sec2(4)==5) then Octet Polar stereographic grid	GDS Content
1	7-8	I Dimension of the grid
2	9-10	J Dimension of the grid
3	11-13	Starting Latitude of the grid.
4	14-16	Starting Longitude of the grid.
5	17	Resolution and component flags.
6	18-20	Center longitude of the projection.
7	21-23	Grid-spacing in the I direction
8	24-26	Grid-spacing in the J direction
9	27	Projection center
10	28	Scanning mode (bit 3 from Table 8)
21	29	Iscan sign (+1/-1) (bit 1 from Table 8)
22	30	Jscan sign (+1/-1) (bit 2 from Table 8)
INFOGRID	if(sec2(4)==50) then Spherical Harmonic Coefficients Octet	GDS Content
1	7-8	J-pentagonal resolution parameter
2	9-10	K-pentagonal resolution parameter
3	11-12	M-pentagonal resolution parameter
4	13	Spectral representation type (ON388 Table 9)
5	14	Coefficient storage mode (ON388 Table 10)
	15-32	Set to 0 (reserved)

## I.11 Additional information for setting up the READ\_GRIB routines for use on Linux Machines

A few steps were taken to modify the original READ\_GRIB routines to make them more compatible with Absoft, Lahey95, and other Linux (32-bit and 64-bit) based compilers. Here is a list of those steps:

Replaced the extensions of each \*.F file with \*.F90.

In the C-routine, cio.c, the following lines of code were added or modified:

- Line 19 Added “|| defined(ABSOFT)and || defined(LAHEY)”

In the Makefile, the following lines of code were added or modified:

- Line 19 Added “.F90” to .SUFFIXES rule
- Line 34-35 – Added “@echo make absoft” and “@echo make lahey”; resp.
- Line 71 (and following lines) Added flags and compiler names for ABSOFT:

**absoft:**

```

$(MAKE) $(LIBTARGET) \
"FC = f90" \
"FCFLAGS = -O -YEXT_NAMES=LCS -B108 -YCFRL=1 -YDEALLOC=ALL - \
DHIDE_SHR_MSG -DNO_SHR_VMATH -DABSOFT -DLITTLE_ENDIAN -DBIT -DBIT32" \
"CC = gcc" \
"CCFLAGS = -O -Wall -DABSOFT -DLITTLE_ENDIAN -DG_ENABLE_DEBUG=1" \
"CPP = /lib/cpp" \
"CPPFLAGS = -C -P -DBIT32"

lahey:
$(MAKE) $(LIBTARGETS) \
"FC = lf95" \
"FCFLAGS = -O -DBIT32 -DLINUX -DLAHEY -DLITTLE_ENDIAN" \
"CC = cc" \
"CCFLAGS = -O -DUSE_GCC -DLAHEY -DLITTLE_ENDIAN" \
"CPP = /lib/cpp" \
"CPPFLAGS = -C -P "

```

- Line 90 Changed “.F.o” to “.F90.o”
- Line 91 Removed “-d” from the rule

## I.12 Example of Fortran code that calls READ\_GRIB routines

```

!-- Initialize certain variables and parameters of GRIB file:
nunit = 10           ! Fortran unit number
ufn = nunit + 1     ! UNIX file descriptor associated with "nunit"
datarray = 0

!-- Open INPUT GRIB File:
call copen (nunit, ufn, trim(input_file)//char(0), 1, iret, 1)
print *, " ** Open File Code: ", iret

if ( iret > 0) then      ! Return File Error Code Number - IF FAILED TO OPEN!
    write(*,*) "STOPPING ROUTINE -- FILE NOT OPENED DUE TO CODE # :: ", iret
    stop
end if

!-- Read GRIB file:
call gribread ( ufn, datarray, ndata, ierr )

if ( ierr > 0) then    ! Return File Error Code Number - IF FAILED TO READ!
    write(*,*) "STOPPING ROUTINE -- FILE NOT READ DUE TO CODE # :: ", ierr

```

```
        stop
      end if

!Print GRIB Header Information
do isec = 0, 2
  call gribprint (isec)
end do

do i = 1, ndata
  if (datarray(i) >0 ) then
    print *, i, datarray(i)
    end if
end do
```

Table 3: SEC0: GRIB Header Section 0 information

Number	Description
1	Length of a complete GRIB record
2	Grib Edition Number

Table 4: SEC1: GRIB Header Section 1 information

Octet	PDS content
1-3	Length of GRIB section 1 (3 bytes)
4	Parameter Table Version number
5	Center Identifier
6	Generating process Identifier
7	Grid ID number for pre-specified grids.
8	Binary bitmap flag:
9	Parameter ID Number and Units (ON388 Table 2)
10	Indicator of level type or layer (ON388 Table 3)
11	Level value (height or pressure), of the top value of a layer
12	Level value, but for bottom value of a layer ( 0 if NA ??)
13	Year (00-99)
14	Month (01-12)
15	Day of the month (01-31)
16	Hour (00-23)
17	Minute (00-59)
18	Forecast time unit: (ON388 Table 4)
19	Time period 1 (Number of Time Units Given in Octet 18)
20	Time period 2 or time interval between successive analyses
21	Time range indicator (ON833 Table 5)
22-23	Number included in average when Octet 21 (Table 5) indicates average or accumulation (otherwise set to 0)
24	Number missing from averages or accumulations
25	Century (Years 1999 and 2000 are century 20, 2001 is century 21)
26	Sub-center identifier
27-28	Decimal scale factor D. Negative value indicates setting high order bit in Octet 27 to 1 (“on”).
29	Is there a GDS (0=no, 1=yes; bit 1 of sec1(6)) Refer to Octet 8 above
30	Is there a BMS (0=no, 1=yes; bit 2 of sec1(6)) Refer to Octet 8 above

## J GRIB Output Information

### Introduction:

This is a package of subroutines to write GRIB data in LIS. As land surface models become more sophisticated, more variable metadata will need to be added. The package currently supports all of the mandatory ALMA output variables for LSMS. However, the GRIB output module can easily be extended to support additional variables. This new LIS grib interface was adapted from a module similair to one used in the AFWA AGRMET model. Supported LIS projections include Lat/Lon, Lambert Conformal, Polar Stereographic, and Mercator. The setup of the GRIB grid description section (GDS section) is handled automatically by LIS within the domain initialization module.

Charles J. Alonge  
SAIC/NASA GSFC  
Winter 2006/2007, and continuing

The user interfaces are:

**SUBROUTINE GRIB1\_SETUP** - Initializes variable independent information in the GRIB product definition section (PDS).

**SUBROUTINE GRIB1\_FINALIZE** - Finalizes the product definition section of the GRIB record by encoding variable specific metadata into the output grib record.

**SUBROUTINE DRV\_WRTIEVAR\_GRIB** - Writes the grib record (and stats) by gathering the variable from the individual MPI tasks (if applicable) and calls the lower level grib output routines

### J.1 SUBROUTINE GRIB1\_SETUP

- Call: GRIB1\_SETUP(SECT1, INFO, BITMAP, DATE)
- Initialize variable independent information in the GRIB product definition record (e.g. center, subcenter, time valid, and bitmap flag)
- NOTE!: BITMAP MUST ALWAYS BE TRUE as the code will handle output over all gridpoints (including water) or just LIS output over land.

Input: SECT1 (integer(36)): GRIB PDS array. Contains variable specific GRIB metadata.

INFO (integer(5)): GRIB packing descriptor, describes the length of each section of grib header for low-level grib encoding.

BITMAP (logical) : Determines if a bitmap is used in the grib encoding (ALWAYS leave defined as true!!)

DATE (character(10)): Character string containing the date of the grib record ( format: YYYYMMDDHH )

Side Effects: NONE - Do not set Bitmap to false or grib packing will fail.

## J.2 SUBROUTINE GRIB1\_FINALIZE

- Call: SUBROUTINE GRIB1\_FINALIZE(GRIB\_INDEX, SECT1, TIME\_UNIT, TIME\_1, TIME\_2, TIME\_RANGE)
- Finalizes the PDS section of a grib record by encoding variable specific metadata into the output grib record.

Input: GRIB\_INDEX (integer): Index into the GRIB pds values array corresponding to the variable being processed for output (See tables below for enumerated types of this value for the different output variables)  
SECT1 (integer (36)): Parameters for Section 1 (PDS) of the GRIB record  
TIME\_UNIT (integer): Units in time for output contained in GRIB record.  
These are encoded as follows:

- 0** - Minute
- 1** - Hour
- 2** - Day
- 3** - Month
- 4** - Year
- 254** - Second

TIME\_1(integer): Time1 for GRIB Record Descriptor. Used only when averaging or accumulating variables over a period of time (See TIME\_RANGE description).

TIME\_2(integer): Time2 for GRIB Record Descriptor. For general LIS usage this should always be set to zero.

TIME\_RANGE: Time Range Indicator. Describes relationship between TIME\_1 and TIME\_2. These are encoded as follows (for a more detailed description please refer to the WMO Grib 1 Manual - Table 5):

- 0** - Forecast product value for reference TIME\_1 (TIME\_2 ignored)
- 1** - Analysis product for reference TIME\_1 (TIME\_2 ignored)
- 2** - Product wth a valid time ranging between +TIME\_1 and +TIME\_2
- 3** - Average (reference time +TIME\_1 to reference time +TIME\_2)
- 4** - Accumulation (reference time +TIME\_1 to reference time +TIME\_2)  
product considered valid at TIME\_2
- 5** - Difference (reference time +TIME\_2 minus reference time +TIME\_1)  
product considered valid at TIME\_2
- 6** - Average (reference time -TIME\_1 to reference time -TIME\_2)
- 7** - Average (reference time -TIME\_1 to reference time +TIME\_2)

Side Effects: Do not use negative values when defining the three time descriptor variables as this will cause an error in the low-level grib packing routines. Instead use a different time range indicator.

### J.3 SUBROUTINE DRV\_WRITEVAR\_GRIB

- Call: DRV\_WRITEVAR\_GRIB(FTN, FTN\_STATS, N, FLAG, VAR, MVAR, FORM, TOPLEV, BOTLEV, KDIM)
- Writes the grib record by gathering the variable from the individual MPI tasks and calling the lower level grib output routines

Input: FTN (integer): Unit number of grib output file.  
 FTN\_STATS (integer): Unit number of grib output file.  
 N (integer): Index of the LIS domain or nest.  
 FLAG (integer): Unit number of grib output file.  
 VAR (real(lis%nch(n))): Variable output data.  
 MVAR (character(len=\*)): Name of variable being written.  
 FORM (integer): Format to be used in stats file (1-decimal,2-scientific)  
 TOPLEV (real(KDIM)): Format to be used in stats file (1-decimal,2-scientific)  
 BOTLEV (real(KDIM)): Format to be used in stats file (1-decimal,2-scientific)  
 KDIM (integer): Grid dimension Format to be used in stats file (1-decimal,2-scientific)  
 INFO (integer(5)): GRIB packing descriptor, describes the length of each section of grib header for low-level grib encoding.  
 BITMAP (logical) : Determines if a bitmap is used in the grib encoding (ALWAYS leave defined as true!!)  
 DATE (character(10)): Character string containing the date of the grib record ( format: YYYYMMDDHH )

Side Effects: NONE - Do not set Bitmap to false or grib packing will fail.

### J.4 Output GRIB Variables

GRIB ID	Table Number	Description
ALMA ENERGY BALANCE COMPONENTS		
GRIB_SWNET	1	Net Shortwave Radiation Flux
GRIB_LWNET	2	Net Longwave Radiation Flux
GRIB_QLE	3	Latent Heat Flux
GRIB_QH	4	Sensible Heat Flux
GRIB_QG	5	Ground Heat Flux
GRIB_QF	6	Energy of Fusion
GRIB_QV	7	Energy of Sublimation
GRIB_QTAU	8	Momentum Flux
GRIB_QA	9	Advection Energy
GRIB_DELSRFHEAT	10	Change in Surface Heat Storage
GRIB_DELCLDCNT	11	Change in Snow Cold Content
ALMA WATER BALANCE COMPONENTS		

GRIB_SNOWF	12	Snowfall Rate
GRIB_RAINF	13	Rainfall Rate
GRIB_EVAP	14	Total Evaporation
GRIB_QS	15	Surface Runoff
GRIB_QREC	16	Recharge
GRIB_QSB	17	Subsurface Runoff
GRIB_QSM	18	Snowmelt
GRIB_QFZ	19	Refreezing of Water in the Snow
GRIB_QST	20	Snow Throughfall
GRIB_DELSM	21	Change in Soil Moisture
GRIB_DELSWE	22	Change in Snow Water Equivalent
GRIB_DELSRFSTR	23	Change in Surface Water Storage
GRIB_DELINTCPT	24	Change in Interception Storage
ALMA SURFACE STATE VARIABLES		
GRIB_SNOWT	25	Snow Surface Temperature
GRIB_VEGT	26	Vegetation Canopy Temperature
GRIB_BARESOILT	27	Temperature of Bare Soil
GRIB_AVGSURFT	28	Average Surface Temperature
GRIB_RADT	29	Surface Radiative Temperature
GRIB_ALBEDO	30	Surface Albedo
GRIB_SWE	31	Snow Water Equivalent (SWE)
GRIB_SWEVEG	32	SWE intercepted by Vegetation
GRIB_SURFSTOR	33	Surface Water Storage
ALMA SUBSURFACE STATE VARIABLES		
GRIB_SOILMOIST	34	Soil Moisture
GRIB_SOILTEMP	35	Soil Temperature
GRIB_LSOILMOIST	36	Avg. layer fraction of Liquid Moisture
GRIB_FSOILMOIST	37	Avg. layer fraction of Frozen Moisture
GRIB_SOILWET	38	Total Soil Wetness
ALMA EVAPORATION COMPONENTS		
GRIB_POTEVAP	39	Potential Evaporation
GRIB_ECANOP	40	Interception Evaporation
GRIB_TVEG	41	Vegetation Transpiration
GRIB_ESOIL	42	Bare Soil Evaporation
GRIB_EWATER	43	Open Water Evaporation
GRIB_ROOTMOIST	44	Root Zone Soil Moisture
GRIB_CANOPINT	45	Total Canopy Water Storage
GRIB_ESNOW	46	Snow Evaporation
GRIB_SUBSNOW	47	Snow Sublimation
GRIB_SUBSURF	48	Sublimation of Snow Free Area
GRIB_ACOND	49	Aerodynamic Conductance
FORCING VARIABLES		
GRIB_WINDFORC	50	Wind Speed
GRIB_RAINFFORC	51	Rainfall Forcing
GRIB_SNOWFFORC	52	Snowfall Forcing

GRIB_TAIRFORC	53	Air Temperature
GRIB_QAIRFORC	54	Specific Humidity
GRIB_PSLANDFORC	55	Surface Pressure
GRIB_SWDOWNFORC	56	Downwelling Shortwave Radiation Flux
GRIB_LWDOWNFORC	57	Downwelling Longwave Radiation Flux
PARAMETER OUTPUT - EXPERIMENTAL		
GRIB_LANDMASK	58	Land Mask
GRIB_LANDCOVER	59	Vegetation Type - Landcover
GRIB_SOILTYPE	60	Soil Type
GRIB_SOILCOLOR	61	Soil Color
GRIB_TOPOGRAPHY	62	Topography of Land Surface
GRIB_LAI	63	Leaf Area Index
GRIB_SAI	64	Stem Area Index
GRIB_SNFRALBEDO	65	Snow-free Albedo
GRIB_MXSNALBEDO	66	Maximum Snow Albedo
GRIB_GREENNESS	67	Greenness Fraction
GRIB_TEMP_BOT	68	Bottom Temperature

## J.5 Example of Fortran code that calls GRIB output routines

```

! Setup of GRIB Metadata Section

! toplev is the depth of the top of each soil layer
! botlev is the depth of the bottom of each soil layer
toplev(1) = 0.0
botlev(1) = noah_struc(n)%lyrthk(1)

! determine bounding levels for each soil moisture layer
do i = 2, noah_struc(n)%nslay
    toplev(i) = toplev(i-1) + noah_struc(n)%lyrthk(i-1)
    botlev(i) = botlev(i-1) + noah_struc(n)%lyrthk(i)
enddo

! Convert to centimeters -- the depths for layers below the land
! surface (surface = 112) are expected in centimeters,
! per GRIB specifications.
toplev = toplev * 100.0
botlev = botlev * 100.0

! Set values for non layered fields (Fluxes, Sfc Fields, etc.)
toplev0 = 0
botlev0 = 0

! Set Date String to Pass to GRIB Module

```

```

hr1=lis%hr
da1=lis%da
mo1=lis%mo
yr1=lis%yr
write(unit=date,fmt='(i4.4,i2.2,i2.2,i2.2)') yr1,mo1,da1,hr1

! Setup common information to go into the PDS section
! of the GRIB file (Variable Independent Metadata)

call grib1_setup(gribobj(n)%sect1, gribobj(n)%info, &
                  gribobj(n)%bitmap, date)

! Set time units of output for GRIB file
! Is output interval in days,hours,minutes,seconds?
! Also, determine output time range in that unit (time_past)

if(noah_struc(n)%outInterval .GT. 0) then
    time_unit = 254      ! seconds
    time_past = (noah_struc(n)%outInterval / 1)
endif
if(noah_struc(n)%outInterval .GE. 60) then
    time_unit = 0        ! minutes
    time_past = (noah_struc(n)%outInterval / 60)
endif
if(noah_struc(n)%outInterval .GE. 3600) then
    time_unit = 1        ! hours
    time_past = (noah_struc(n)%outInterval / 3600)
endif
if(noah_struc(n)%outInterval .GE. 86400) then
    time_unit = 2        ! days
    time_past = (noah_struc(n)%outInterval / 86400)
endif

! End of GRIB metadata section

! Sample Outputs.....

! Output Net Shortwave Radiation

! Finalize PDS section of this variable, in this case SWNET is a time
! averaged variable => time range indicator equals 7
call grib1_finalize(GRIB\_SWNET,gribobj(n)%sect1,time_unit,time_past,0,7)

noah_struc(n)%noah%swnet = noah_struc(n)%noah%swnet/float(noah_struc(n)%count)

! Call low level grib routines (single level data, hence 1 as final arg)

```

```

call drv_writevar_grib(ftn,ftn_stats,n,metadata_output%swnet,      &
noah_struc(n)%noah%swnet,"Swnet(W/m2)",1, &
toplev0, botlev0, 1)

! Output Soil Moisture

! Store soil moisture in a 2D array (LSM points x Num. Layers)
temp_nslay = 0.0
do i=1,noah_struc(n)%nslay
  do t=1,lis%nch(n)
    temp_nslay(t,i) = noah_struc(n)%noah(t)%soilmoist(i)
  enddo
enddo

! Finalize PDS section of this variable, in this case SOILMOIST is an
! instantaneous variable => time range indicator equals 1 - analysis
call grib1_finalize(GRIB\_SOILMOIST,gribobj(n)%sect1,time_unit,0,0,1)

! Call low level grib routines (output all levels, number of layers
! is used as the last argument)
call drv_writevar_grib(ftn,ftn_stats,n,metadata_output%soilmoist,&
temp_nslay,"SoilMoist(kg/m2)",2,           &
toplev, botlev, noah_struc(n)%nslay)

```

## References

- [1] GrADS. <http://grads.iges.org/grads/grads.html>.
- [2] Protex documenting system. <http://gmao.gsfc.nasa.gov/software/protex>.
- [3] CLM. <http://www.cgd.ucar.edu/tss/clm>.
- [4] DODS. <http://www.unidata.ucar.edu/packages/dods/>.
- [5] NOAH. <ftp://ftp.ncep.noaa.gov/pub/gcp/lidas/noahlsm/>.